

E 5-64

MENU

SEARCH

INDEX

DETAIL

1/1



JAPANESE PATENT OFFICE

PATENT ABSTRACTS OF JAPAN

(11)Publication number: 09128346

(43)Date of publication of application: 16.05.1997

(51)Int.Cl.

G06F 15/16

G06F 13/36

G06F 15/163

(21)Application number: 07285829

(71)Applicant:

MITSUBISHI ELECTRIC CORP

(22)Date of filing: 02.11.1995

(72)Inventor:

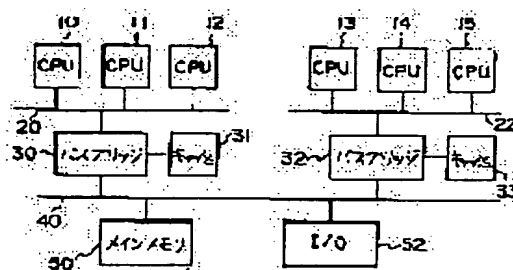
KAMEMARU TOSHIHISA
YASUNAGA HIROAKI

(54) HIERARCHICAL BUS SYSTEM

(57)Abstract:

PROBLEM TO BE SOLVED: To increase the number of CPU's which can be connected to a system in the multiprocessor system.

SOLUTION: CPU's 10-12 are connected to a bus bridge 30 through a host bus 20 and CPU 13-15 are connected to a bus bridge 32 through a host bus 22. The bus bridges 30 and 32 are connected to a main memory 50 and an I/O device 52 through a slave bus 40. Bridge caches 31 and 33 shared by host CPU 10-12 and 13-15 are connected to the bus bridges 30 and 32. The host buses 20 and 22 are split buses and CPU's 10-15 can deal with the split. The number of CPU's which can be connected is increased by hierarchization with such constitution, and the deterioration of throughput and latency, accompanying hierarchization, can be prevented by the adoption of the bridge cache and by making the



BEST AVAILABLE COPY

host buses into the split buses.

LEGAL STATUS

[Date of request for examination]

[Date of sending the examiner's decision of rejection]

[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]

[Date of final disposal for application]

[Patent number]

[Date of registration]

[Number of appeal against examiner's decision of rejection]

[Date of requesting appeal against examiner's decision of rejection]

[Date of extinction of right]

Copyright (C); 1998 Japanese Patent Office

MENU

SEARCH

INDEX

DETAIL

E5264

(19)日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11)特許出願公開番号

特開平9-128346

(43)公開日 平成9年(1997)5月16日

(51)Int.Cl. ⁸	識別記号	庁内整理番号	F I	技術表示箇所
G 0 6 F 15/16			G 0 6 F 15/16	4 0 0 B
13/36	3 1 0		13/36	3 1 0 E
15/163			15/16	3 2 0 K

審査請求 未請求 請求項の数42 O L (全 48 頁)

(21)出願番号 特願平7-285829

(22)出願日 平成7年(1995)11月2日

(71)出願人 000006013

三菱電機株式会社

東京都千代田区丸の内二丁目2番3号

(72)発明者 亀丸 敏久

東京都千代田区丸の内二丁目2番3号 三菱電機株式会社内

(72)発明者 安永 裕明

東京都千代田区丸の内二丁目2番3号 三菱電機株式会社内

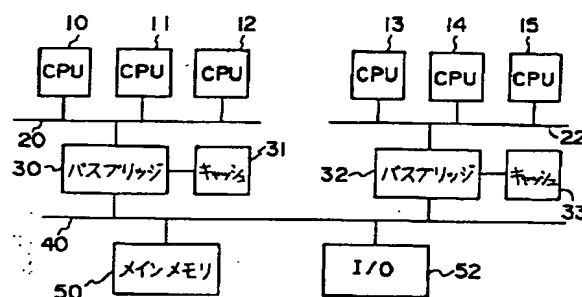
(74)代理人 弁理士 吉田 研二 (外2名)

(54)【発明の名称】 階層バスシステム

(57)【要約】

【課題】 マルチプロセッサシステムにおいて、システムに接続可能なCPUの数を増加させる。

【解決手段】 CPU10～12は上位バス20を介してバスブリッジ30に接続され、CPU13～15は上位バス22を介してバスブリッジ32に接続される。バスブリッジ30及び32とが下位バス40を介してメインメモリ50及びI/O装置52が接続される。バスブリッジ30及び32にはそれぞれ上位のCPU10～12及び13～15によって共有されるブリッジキャッシュ31及び33が接続される。上位バス20及び22はスプリットバスであり、CPU10～15はスプリット対応可能となっている。この構成によれば、階層化により接続可能なCPU数が増えるとともに、ブリッジキャッシュの採用及び上位バスのスプリットバス化により、階層化に伴うスループットやレイテンシの劣化を防止することができる。



【特許請求の範囲】

【請求項1】 複数のプロセッサとそれらプロセッサによって共有されるメインメモリとを含む階層バスシステムにおいて、

前記プロセッサを複数個ずつ上位バスにて相互に接続し、

各上位バスごとに、当該上位バスに接続されるプロセッサによって共有されるブリッジキャッシュを有するバスブリッジを設け、

各バスブリッジを下位バスにて相互に接続し、この下位バスに前記メインメモリを接続し、

前記プロセッサをスプリット対応可能とし、

前記プロセッサから上位バスに発行されたトランザクションを前記バスブリッジにてスプリット処理することを特徴とする階層バスシステム。

【請求項2】 請求項1記載の階層バスシステムにおいて、

前記プロセッサは、各々プロセッサキャッシュを有し、前記プロセッサキャッシュ及びブリッジキャッシュは、各キャッシュブロックのステートを記憶するステート記憶手段をそれぞれ有し、

前記プロセッサ及びバスブリッジは、自らが有するプロセッサキャッシュ又はブリッジキャッシュの前記ステート記憶手段を制御するステート制御手段をそれぞれ有し、

前記プロセッサキャッシュの各キャッシュブロックの取り得るステートを、前記ブリッジキャッシュの対応キャッシュブロックのステートに基づいて制約することにより、ブリッジキャッシュとこのブリッジキャッシュの上位のプロセッサキャッシュとのマルチレベル包含性を維持することを特徴とする階層バスシステム。

【請求項3】 請求項2記載の階層バスシステムであって、前記プロセッサキャッシュ及びブリッジキャッシュのステートがMESIプロトコルに従う場合において、ブリッジキャッシュのステートがIの場合には、上位のプロセッサキャッシュの取り得るステートをIに限定し、

ブリッジキャッシュのステートがSの場合には、上位のプロセッサキャッシュの取り得るステートをS又はIに限定し、

ブリッジキャッシュのステートがEの場合には、上位のプロセッサキャッシュの取り得るステートをS又はIに限定し、

ブリッジキャッシュのステートがMの場合には、上位のプロセッサキャッシュがすべてのステートを取り得るように認めることを特徴とする階層バスシステム。

【請求項4】 請求項2記載の階層バスシステムであって、前記プロセッサキャッシュ及びブリッジキャッシュのステートがMESIプロトコルに従う場合において、ブリッジキャッシュのステートがIの場合には、上位の

プロセッサキャッシュの取り得るステートをIに限定し、

ブリッジキャッシュのステートがSの場合には、上位のプロセッサキャッシュの取り得るステートをS又はIに限定し、

ブリッジキャッシュのステートがE又はMの場合には、上位のプロセッサキャッシュがすべてのステートを取り得るように認めることを特徴とする階層バスシステム。

【請求項5】 請求項3又は請求項4記載の階層バスシステムにおいて、

前記バスブリッジは、上位バスから受け取ったトランザクションの種類と当該トランザクションのアドレスに対応するブリッジキャッシュのステートとに基づき、当該トランザクションをスプリットするか否か及びスプリットする場合において下位バスに発行するトランザクションを決定する下位バス出力判定手段を有し、上位バスからトランザクションを受け取った場合、この下位バス出力判定手段によって決定されたトランザクションを下位バスに発行することを特徴とする階層バスシステム。

【請求項6】 請求項5に記載の階層バスシステムにおいて、

前記下位バス出力判定手段は、

上位バスのトランザクションがリードである場合には、ブリッジキャッシュの対応ブロックのステートがIである場合にのみ当該トランザクションをスプリットして下位バスにリード・トランザクションを発行し、

上位バスのトランザクションがインバリデートである場合には、ブリッジキャッシュの対応ブロックのステートがSである場合にのみ当該トランザクションをスプリットして下位バスにインバリデート・トランザクションを発行し、

上位バスのトランザクションがリード・アンド・インバリデートである場合、ブリッジキャッシュの対応ブロックのステートがSである場合には当該トランザクションをスプリットして下位バスにインバリデート・トランザクションを発行し、ブリッジキャッシュの対応ブロックのステートがIである場合には当該トランザクションをスプリットして下位バスにリード・アンド・インバリデート・トランザクションを発行するように決定することを特徴とする階層バスシステム。

【請求項7】 請求項6記載の階層バスシステムにおいて、

前記下位バス出力判定手段は、

上位バスのトランザクションがリードである場合に、当該リード・トランザクションが命令コードを要求するものであるかデータを要求するものであるかを判別する要求判別手段を有し、

当該リード・トランザクションがデータを要求するものである場合には下位バスにリード・アンド・インバリデートを発行することを特徴とする階層バスシステム。

【請求項8】 請求項3又は請求項4に記載の階層バスシステムにおいて、

前記バスブリッジは、下位バスに発行したトランザクションの種類と下位バスのスヌープ結果に基づき、ブリッジキャッシュにおける対応キャッシュブロックの次ステートを制御する第1のステート制御手段を有することを特徴とする階層バスシステム。

【請求項9】 請求項8記載の階層バスシステムにおいて、

前記第1のステート制御手段は、

前記トランザクションの種類がインバリデート又はリード・アンド・インバリデートである場合は、前記ブリッジキャッシュの対応ブロックの次ステートをMにし、
前記トランザクションの種類がリードである場合は、下位バスのスヌープ結果がミスの場合には前記ブリッジキャッシュの対応ブロックの次ステートをEにし、下位バスのスヌープ結果がHIT又はHITMの場合には前記ブリッジキャッシュの対応ブロックの次ステートをSにすることを特徴とする階層バスシステム。

【請求項10】 請求項3記載の階層バスシステムにおいて、

前記バスブリッジは、

自バスブリッジから下位バスに発行したトランザクションの種類に基づきスプリット応答時における上位バスに対するスヌープ出力を生成する上位バススヌープ出力生成手段を有し、

前記上位バススヌープ出力生成手段は、前記トランザクションの種類がインバリデート又はリード・アンド・インバリデートである場合は前記スヌープ出力をミスとし、前記トランザクションの種類がリードである場合は前記スヌープ出力をHITとし、
バスブリッジが上位バスのトランザクションをスプリットして下位バスにトランザクションを発行した場合に、前記上位バススヌープ出力生成手段からのスヌープ出力によって、上位バスを介して当該バスブリッジに接続されたプロセッサのプロセッサキャッシュのステートを制御することを特徴とする階層バスシステム。

【請求項11】 請求項4記載の階層バスシステムにおいて、

前記バスブリッジは、

自バスブリッジから下位バスに発行したトランザクションの種類及び当該トランザクションに対する下位バスからのスヌープ結果に基づき、スプリット応答時における上位バスに対するスヌープ出力を生成する上位バススヌープ出力生成手段を有し、

前記上位バススヌープ生成手段は、前記トランザクションの種類がインバリデート又はリード・アンド・インバリデートである場合は前記スヌープ出力をミスとし、前記トランザクションの種類がリードである場合は、下位バスのスヌープ結果がミスの場合は前記スヌープ出力を

ミスとし下位バスのスヌープ結果がHIT又はHITMの場合は前記スヌープ出力をHITとし、

バスブリッジが上位バスのトランザクションをスプリットして下位バスにトランザクションを発行した場合に、前記上位バススヌープ出力生成手段からのスヌープ出力によって、上位バスを介して当該バスブリッジに接続されたプロセッサのプロセッサキャッシュのステートを制御することを特徴とする階層バスシステム。

【請求項12】 請求項3に記載の階層バスシステムにおいて、

前記バスブリッジは、

下位バスから受け取ったトランザクションの種類と当該トランザクションのアドレスに対応するブリッジキャッシュのステートとに基づき、上位バスに発行するトランザクションの種類を決定する上位バス出力判定手段を有し、

前記上位バス出力判定手段は、前記トランザクションの種類がリードである場合はブリッジキャッシュの対応ステートがMの場合に上位バスにリード・トランザクションを発行すると決定し、前記トランザクションの種類がインバリデート又はリード・アンド・インバリデートである場合は前記ブリッジキャッシュの対応ステートがM又はE又はSの場合に上位バスにインバリデート・トランザクションを発行すると決定することを特徴とする階層バスシステム。

【請求項13】 請求項4に記載の階層バスシステムにおいて、

前記バスブリッジは、

下位バスから受け取ったトランザクションの種類と当該トランザクションのアドレスに対応するブリッジキャッシュのステートとに基づき、上位バスに発行するトランザクションの種類を決定する上位バス出力判定手段を有し、

前記上位バス出力判定手段は、前記トランザクションがリードである場合はブリッジキャッシュの対応ステートがM又はEの場合に上位バスにリード・トランザクションを発行すると決定し、前記トランザクションの種類がインバリデート又はリード・アンド・インバリデートである場合は前記ブリッジキャッシュの対応ステートがM又はE又はSの場合に上位バスにインバリデート・トランザクションを発行すると決定することを特徴とする階層バスシステム。

【請求項14】 請求項12又は請求項13記載の階層バスシステムにおいて、

前記バスブリッジは、

ブリッジキャッシュの各キャッシュブロックごとについて、当該キャッシュブロックについての所定のトランザクションを上位バスに対して発行する必要があるか否かの状態を示す状態フラグを記憶した上位キャッシュ状態フラグ記憶手段と、

上位バスに発行されたトランザクションをモニタして、当該トランザクションの種類及びアドレスに基づいて前記上位キャッシュ状態フラグ記憶手段の当該キャッシュブロックの状態を制御するフラグ制御手段と、

下位バスから前記所定トランザクションを受け取った場合に、前記上位キャッシュ状態フラグを参照し、当該所定トランザクションのアドレスに対応するキャッシュブロックの状態フラグが上位バスに対して当該所定トランザクションを発行する必要がないことを示している場合は、当該所定トランザクションの上位バスへの発行を抑制する発行抑制手段と、

を有することを特徴とする階層バスシステム。

【請求項15】 請求項3記載の階層バスシステムにおいて、

前記バスブリッジは、

前記下位バスから受け取ったトランザクションの種類と、当該トランザクションのアドレスについてのブリッジキャッシュのステートとに基づき、前記下位バスに対するスヌープ出力を生成する下位バススヌープ出力生成手段を有し、

前記下位バススヌープ出力生成手段は、

前記トランザクションの種類がリードの場合において、前記ブリッジキャッシュの対応ステートがMの場合はHITMをスヌープ出力とし、前記ブリッジキャッシュの対応ステートがE又はSの場合はHITをスヌープ出力とし、

前記トランザクションの種類がインバリデート又はリード・アンド・インバリデートの場合には、前記ブリッジキャッシュの対応ステートがMの場合にHITMをスヌープ出力とすることを特徴とする階層バスシステム。

【請求項16】 請求項4記載の階層バスシステムにおいて、

前記バスブリッジは、

前記下位バスから受け取ったトランザクションの種類と、当該トランザクションのアドレスについてのブリッジキャッシュのステートと、当該トランザクションに対応して上位バスに発行したトランザクションに対する上位バスのスヌープ結果とに基づき、前記下位バスに対するスヌープ出力を生成する下位バススヌープ出力生成手段を有し、

前記下位バススヌープ出力生成手段は、

前記トランザクションの種類がリードの場合において、前記スヌープ結果がHITMの場合は前記ブリッジキャッシュの対応ステートがM又はEの場合にHITMをスヌープ出力とし、前記スヌープ結果がHITM以外の場合は前記ブリッジキャッシュの対応ステートがMの場合にはHITMをスヌープ出力とし前記ブリッジキャッシュの対応ステートがE又はSの場合にはHITをスヌープ出力とし、

前記トランザクションの種類がインバリデート又はリー

ド・アンド・インバリデートの場合において、前記スヌープ結果がHITMの場合は前記ブリッジキャッシュの対応ステートがM又はEの場合にHITMをスヌープ出力とし、前記スヌープ結果がHITM以外の場合は前記ブリッジキャッシュの対応ステートがMの場合にHITMをスヌープ出力とすることを特徴とする階層バスシステム。

【請求項17】 請求項3又は請求項4に記載の階層バスシステムにおいて、

前記バスブリッジは、下位バスから受け取ったトランザクションに対応して上位バスに発行したトランザクションの種類に基づき、ブリッジキャッシュの対応ブロックの次ステートを制御する第2のステート制御手段を有し、

前記第2のステート制御手段は、

前記トランザクションの種類がリードである場合は、前記ブリッジキャッシュの対応ブロックの次ステートをSにし、

前記トランザクションの種類がインバリデート又はリード・アンド・インバリデートである場合は、前記ブリッジキャッシュの対応ブロックの次ステートをIにすることを特徴とする階層バスシステム。

【請求項18】 請求項3又は請求項4に記載の階層バスシステムにおいて、

前記バスブリッジは、下位バスから受け取ったトランザクションに対応して上位バスに発行したトランザクションの種類と、当該トランザクションに対する上位バスのスヌープ結果とに基づき、ブリッジキャッシュの対応ブロックの次ステートを制御する第2のステート制御手段を有し、

前記第2のステート制御手段は、

前記トランザクションの種類がリードである場合は、前記上位バスのスヌープ結果がミスの場合には前記ブリッジキャッシュの対応ブロックの次ステートをIにし、前記上位バスのスヌープ結果がHIT又はHITMの場合には前記ブリッジキャッシュの対応ブロックの次ステートをSにし、

上位バスに発行したトランザクションがインバリデート又はリード・アンド・インバリデートである場合は、前記ブリッジキャッシュの対応ブロックの次ステートをIにすることを特徴とする階層バスシステム。

【請求項19】 請求項3又は請求項4記載の階層バスシステムにおいて、

前記バスブリッジは、

上位バスから受け取ったトランザクションのアドレスに基づき、ブリッジキャッシュにおいて当該アドレスに対応するキャッシュブロックのエビクションが必要か否かを判定するエビクション判定手段と、

エビクションが必要と判定された場合に、前記ブリッジキャッシュのステート記憶手段に記憶されたエビクシ

ン対象のキャッシュブロックのステートに基づき、上位バスに対して前記トランザクションのアドレスに対するインバリデートトランザクションを発行するか否かを判定するインバリデート判定手段であって前記エビクション対象のキャッシュブロックのステートがM又はE又はSの場合に上位バスに対してインバリデート・トランザクションを発行すると判定するインバリデート判定手段と、

を有することを特徴とする階層バスシステム。

【請求項20】 請求項3記載の階層バスシステムにおいて、

前記バスブリッジは、

上位バスから受け取ったトランザクションのアドレスに基づき、ブリッジキャッシュにおいて当該アドレスに対応するキャッシュブロックのエビクションが必要か否かを判定するエビクション判定手段と、

エビクションが必要と判定された場合に、前記ステート記憶手段に記憶されたエビクション対象のキャッシュブロックのステートに基づき下位バスに前記トランザクションのアドレスに対するライトバックトランザクションを発行するか否かを判定するライトバック判定手段であって、エビクション対象のキャッシュブロックのステートがMの場合のみ下位バスにライトバックトランザクションを発行すると判定するライトバック判定手段と、

を有することを特徴とする階層バスシステム。

【請求項21】 請求項4記載の階層バスシステムにおいて、

前記バスブリッジは、

上位バスから受け取ったトランザクションのアドレスに基づき、ブリッジキャッシュにおいて当該アドレスに対応するキャッシュブロックのエビクションが必要か否かを判定するエビクション判定手段と、

エビクションが必要と判定された場合に、上位バスに対して前記キャッシュブロックに対するインバリデートトランザクションを発行するインバリデート発行手段と、エビクションが必要と判定された場合に、前記ステート記憶手段に記憶されたエビクション対象のキャッシュブロックのステートと前記インバリデート発行手段から発行されたインバリデートトランザクションに対する上位バスのスヌープ結果とに基づき下位バスに前記エビクション対象のキャッシュブロックに対するライトバックトランザクションを発行するか否かを判定するライトバック判定手段と、

を有し、

前記ライトバック判定手段は、上位バスのスヌープ結果がHITMの場合はエビクション対象のキャッシュブロックのステートがM又はEのときにライトバックを発行すると判定し、上位バスのスヌープ結果がHITM以外の場合はエビクション対象のキャッシュブロックのステートがMのときにライトバックを発行すると判定するこ

とを特徴とする階層バスシステム。

【請求項22】 請求項19～請求項21のいずれかに記載の階層バスシステムにおいて、

前記バスブリッジは、

ブリッジキャッシュの前記エビクション対象のキャッシュブロックのデータを格納するバッファ手段を有し、エビクション判定手段にてエビクションが必要と判定されると、ブリッジキャッシュのエビクション対象のキャッシュブロックのデータを前記バッファ手段に退避させたのちブリッジキャッシュの当該キャッシュブロックを用いて前記上位バスからのトランザクションに対する処理を実行し、前記バッファ手段に退避したデータを用いてライトバック処理を行うことを特徴とする階層バスシステム。

【請求項23】 請求項22記載の階層バスシステムにおいて、

前記バッファ手段は、

エビクション対象のキャッシュブロックのアドレスを格納するアドレスレジスタと、

エビクション対象のキャッシュブロックのデータを格納するデータバッファと、

を有することを特徴とする階層バスシステム。

【請求項24】 請求項22記載の階層バスシステムにおいて、

前記バッファ手段は、

トランザクションのアドレス指定によってアクセス可能な補助キャッシュとして構成されることを特徴とする階層バスシステム。

【請求項25】 請求項22記載の階層バスシステムにおいて、

前記バスブリッジは、

エビクション判定に応じて当該バスブリッジから上位バスに発行したインバリデートトランザクションに対して、上位バスのプロセッサから該当キャッシュブロックのデータがライトバックされてきた場合に、プロセッサからライトバックされてきたデータによって前記バッファ手段に格納されているデータを更新することを特徴とする階層バスシステム。

【請求項26】 請求項1記載の階層バスシステムにおいて、

前記バスブリッジは、

当該バスブリッジにおいてスプリット中のトランザクションの有無を示すスプリットフラグ手段と、

上位バスから受け取ったトランザクションの種類が、キャッシュブロックを使用する種類であるか否かを判定するトランザクション判定手段と、

上位バスから受け取ったトランザクションのリトライの要否を判定するリトライ判定手段であって、前記スプリットフラグ手段がスプリット中のトランザクションが有ることを示し、かつ上位バスから受け取ったトランザク

ションが前記トランザクション判定手段によってキャッシュブロックを使用する種類であると判定された場合にリトライが必要と判定するリトライ判定手段と、
前記リトライ判定手段によってリトライが必要と判定されたときに、上位バスに対して前記トランザクションについてのリトライ信号を出力するリトライ出力手段と、
を有し、

前記上位バスのトランザクションを発行したプロセッサは、バスブリッジからリトライ信号を受け取った場合に、当該トランザクションをいったん終了し、所定時間後に前記トランザクションを再発行することを特徴とする階層バスシステム。

【請求項27】 請求項1記載の階層バスシステムにおいて、

前記バスブリッジは、

当該バスブリッジにおいてスプリット中のトランザクションが使用するキャッシュブロックのアドレス情報を記憶するスプリット情報記憶手段と、

上位バスから受け取ったトランザクションのアドレスと前記スプリット情報記憶手段に記憶されたアドレス情報とを比較し、上位バスから受け取ったトランザクションがスプリット中のトランザクションの使用するキャッシュブロックを使用するか否かを判定するアドレス判定手段と、

上位バスから受け取ったトランザクションのリトライの要否を判定するリトライ判定手段であって、前記アドレス判定手段において上位バスから受け取ったトランザクションがスプリット中のトランザクションの使用するキャッシュブロックを使用すると判定された場合に、リトライが必要と判定するリトライ判定手段と、
を有し、

前記上位バスのトランザクションを発行したプロセッサは、バスブリッジからリトライ信号を受け取った場合に、当該トランザクションをいったん終了し、所定時間後に前記トランザクションを再発行することを特徴とする階層バスシステム。

【請求項28】 請求項21に記載の階層バスシステムにおいて、

前記バスブリッジは、

上位バスから受け取ったトランザクションのアドレスと、バッファ手段に退避中のキャッシュブロックのアドレスとを比較し、当該トランザクションが退避中のキャッシュブロックに対するトランザクションであるか否かを判定するアドレス比較手段と、

前記上位バスからのトランザクションが退避中のキャッシュブロックに対するトランザクションだと判定された場合に、当該トランザクションに対してリトライが必要と判定するリトライ判定手段と、

前記リトライ判定手段によってリトライが必要と判定されたときに、上位バスに対して前記トランザクションに

ついてのリトライ信号を出力するリトライ出力手段と、
を有し、

前記上位バスのトランザクションを発行したプロセッサは、バスブリッジからリトライ信号を受け取った場合に、当該トランザクションをいったん終了し、所定時間後に前記トランザクションを再発行することを特徴とする階層バスシステム。

【請求項29】 請求項1記載の階層バスシステムにおいて、

前記バスブリッジは、

上位バスから受け取ったトランザクションをスプリットして下位バスにトランザクションを発行した場合において、当該下位バスのトランザクションに対して下位バスからリトライ信号を受け取った場合には、前記上位バスに対してスプリット応答トランザクションを発行すると共に前記上位バスのトランザクション発行元のプロセッサに対してリトライ信号を出力し、

前記トランザクション発行元のプロセッサは、前記バスブリッジからのリトライ信号を受信すると、前記トランザクションをいったん終了し、所定時間後にリトライすることを特徴とする階層バスシステム。

【請求項30】 請求項1記載の階層バスシステムにおいて、

前記バスブリッジは、

上位バストランザクションが有するスプリット識別子に対して下位バスにおいて当該バスブリッジに固有のブリッジ識別子を付加することにより、下位バストランザクションのスプリット識別子を生成するスプリット識別子拡張手段を有し、

上位バスから受け取った上位バストランザクションをスプリットして下位バスに下位バストランザクションを発行する場合に、前記スプリット識別子拡張手段によって得られたスプリット識別子を下位バストランザクションに付することを特徴とする階層バスシステム。

【請求項31】 請求項1記載の階層バスシステムにおいて、

前記バスブリッジは、

一方のバスから受け取ったトランザクションのスプリット識別子を所定ビット数のブリッジトランザクション識別子に変換し、他方のバスにおいて当該バスブリッジに固有なブリッジ識別子に前記ブリッジトランザクション識別子を付加することにより、他方のバスに発行するトランザクションに付与するスプリット識別子を生成するスプリット識別子生成手段を有し、

前記ブリッジトランザクション識別子のビット数と前記ブリッジ識別子のビット数との和は前記一方のバスにおけるスプリット識別子のビット数に等しく、

一方のバスから受け取ったトランザクションに対し他方のバスにトランザクションを発行する必要がある場合には、当該他方のバスに発行するトランザクションに対し

て前記スプリット識別子生成手段によって得られたスプリット識別子を当該他方のバスに発行するトランザクションに付することを特徴とする階層バスシステム。

【請求項32】 請求項31記載の階層バスシステムにおいて、

前記プロセッサは、上位バスにトランザクションを発行する場合に、当該プロセッサが接続された上位バスにおいて当該プロセッサに固有なプロセッサ識別子に対して、当該プロセッサ内において当該トランザクションに固有なプロセッサトランザクション識別子を付加することにより、当該トランザクションのスプリット識別子を生成し、

前記ブリッジ識別子を前記プロセッサ識別子と同ビット数としたことを特徴とする階層バスシステム。

【請求項33】 請求項32記載の階層バスシステムにおいて、

前記バスブリッジのスプリット識別子生成手段は、当該バスブリッジが下位バスに発行するトランザクションに対して付与可能なブリッジトランザクション識別子の使用・未使用状態を管理する下位バス用管理テーブルを有し、

上位バスからのトランザクションに対して下位バスにトランザクションの発行が必要な場合は、前記下位バス用管理テーブルにおいて未使用のブリッジトランザクション識別子を求め、求められたブリッジトランザクション識別子を用いて前記下位バスに発行するトランザクションに対するスプリット識別子を生成するとともに、前記上位バスからのトランザクションのスプリット識別子を当該ブリッジトランザクション識別子に対応づけて前記下位バス用管理テーブルに登録することを特徴とする階層バスシステム。

【請求項34】 請求項33記載の階層バスシステムにおいて、

前記バスブリッジは、上位バスから受け取ったトランザクションに対して下位バスにトランザクションの発行が必要な場合において、前記下位バス用管理テーブルに未使用のブリッジトランザクション識別子がない場合は、上位バスに対してリトライ終了信号を発行し、当該上位バスのトランザクションの発行元プロセッサに対して当該トランザクションのリトライ処理を行わせることを特徴とする階層バスシステム。

【請求項35】 請求項32記載の階層バスシステムにおいて、

前記バスブリッジのスプリット識別子生成手段は、当該バスブリッジが上位バスに発行するトランザクションに対して付与可能なブリッジトランザクション識別子の使用・未使用状態を管理する上位バス用管理テーブルを有し、

下位バスからのトランザクションに対して上位バスにトランザクションの発行が必要な場合は、前記上位バス用

管理テーブルにおいて未使用のブリッジトランザクション識別子を求め、求められたブリッジトランザクション識別子を用いて前記上位バスに発行するトランザクションに対するスプリット識別子を生成するとともに、前記下位バスからのトランザクションのスプリット識別子を当該ブリッジトランザクション識別子に対応づけて前記上位バス用管理テーブルに登録することを特徴とする階層バスシステム。

【請求項36】 請求項35記載の階層バスシステムにおいて、

前記バスブリッジは、下位バスから受け取ったトランザクションに対して上位バスにトランザクションの発行が必要な場合において、前記上位バス用管理テーブルに未使用のブリッジトランザクション識別子がない場合は、前記上位バス用管理テーブルに未使用のブリッジトランザクション識別子ができるのを待って上位バスに対するトランザクションの発行を行うことを特徴とする階層バスシステム。

【請求項37】 請求項1記載の階層バスシステムにおいて、

バスの物理的仕様及び論理的仕様を前記下位バスと上位バスとで共通としたことを特徴とする階層バスシステム。

【請求項38】 請求項37記載の階層バスシステムにおいて、

前記下位バスは、1個以上のボードインタフェースを有し、

これら各ボードインタフェースに対して、1つのプロセッサを実装した第1のボードと、複数のプロセッサ及び上位バス及びバスブリッジを実装した第2のボードと、のいずれかを接続することによりシステム構成を変更可能としたことを特徴とする階層バスシステム。

【請求項39】 請求項1記載の階層バスシステムにおいて、

前記下位バスを複数設け、これら各下位バスにそれぞれメインメモリを接続し、前記各上位バスに対して下位バスと同数のバスブリッジを接続し、前記各上位バスと各下位バスとを互いに1つのバスブリッジを介して接続したことを特徴とする階層バスシステム。

【請求項40】 請求項39記載の階層バスシステムにおいて、

前記複数の下位バスのそれぞれに対してバスブリッジを設け、これらバスブリッジを介して前記各下位バスを1つのI/Oバスに接続し、このI/OバスにI/O装置を接続したことを特徴とする階層バスシステム。

【請求項41】 請求項39又は請求項40に記載の階層バスシステムにおいて、

同一の上位バスに接続された各バスブリッジは、それぞれ受け持ちのアドレス範囲が定められており、上位バスから受け取ったトランザクションのアドレスが自らの受

け持ちアドレス範囲に含まれる場合にのみ、当該トランザクションに対する処理を行うことを特徴とする階層バスシステム。

【請求項42】 請求項41記載の階層バスシステムにおいて、

同一の上位バスに接続された各バスブリッジのアドレス受け持ち範囲を変更可能としたことを特徴とする階層バスシステム。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】本発明は、複数のプロセッサ及びメモリを含むマルチプロセッサシステムにおける階層的なバスシステムに関する。

【0002】

【従来の技術】情報処理システムの処理能力に対する要求の増大に伴い、1つのプロセッサでは能力的に不足する場合が出てきている。このような問題に対処するため、複数のプロセッサを連結して並列動作させるマルチプロセッサ方式のシステムが研究・開発されている。

【0003】マルチプロセッサシステムにおけるプロセッサの結合方式としては従来より様々な方式が考案されているが、実現の容易さや実行するプログラムのプログラミングの容易さなどの理由から、近年ではバス共有型の密結合マルチプロセッサ方式が多く採用されている。図20は、バス共有型の密結合マルチプロセッサシステムを示す概略図である。図20のシステムでは、複数のCPU10～12を、バス90を介してメインメモリ50やI/O装置52に共通接続している。

【0004】

【発明が解決しようとする課題】マルチプロセッサシステムにおいて性能の向上を図る場合、プロセッサ(CPU)の並列数を増やすという方法が考えられるが、バス共有型の場合、1本のバスに接続し得るCPUの数には物理的な制限があった。すなわち、1本のバス上に接続するCPUの数を増やすと信号線の電氣的な容量負荷の増大や信号線の長さの増加によって反射等の問題が発生するので、バス共有型でしかも各CPUを高速動作させようとした場合、並列可能なCPUの数には制限があった。また、CPU数の増加は、いわゆるバスコンテンションの問題を引き起こすため、この点からもCPU数は制限されていた。

【0005】本発明は、このような問題を解決するためになされたものであり、多数のCPUを並列可能な階層バスシステムを提供することを目的とする。

【0006】

【課題を解決するための手段】前述の目的を達成するために、本発明に係る階層バスシステムは、複数のプロセッサとそれらプロセッサによって共有されるメインメモリとを含む階層バスシステムにおいて、前記プロセッサを複数個ずつ上位バスにて相互に接続し、各上位バスご

とに、当該上位バスに接続されるプロセッサによって共有されるブリッジキャッシュを有するバスブリッジを設け、各バスブリッジを下位バスにて相互に接続し、この下位バスに前記メインメモリを接続し、前記プロセッサをスプリット対応可能とし、前記プロセッサから上位バスに発行されたトランザクションを前記バスブリッジにてスプリット処理することを特徴とする。

【0007】この構成では、まず第1に、プロセッサとメインメモリとの間が上位バス及び下位バスからなる階層的なバス構成となっているため、各上位バスに接続されるプロセッサの数が限られる場合でも、上位バスの数を増やすことによりシステム全体として多数のプロセッサを接続することができる。

【0008】第2に、この構成では、各バスブリッジに対して当該バスブリッジの上位バスに接続されるプロセッサによって共有されるブリッジキャッシュを設けたことにより、プロセッサから発行されたトランザクションの要求データがブリッジキャッシュに存在する場合には、下位バスを介してメインメモリを読みに行く必要がなくなるため、下位バスのバス負荷を低減し、下位バスのスループットを向上させることができるとともに、プロセッサがトランザクションに対するレスポンスを得るまでの遅れ時間(レイテンシ)を短くすることができる。

【0009】第3に、この構成では、上位バスをスプリットバスとしたことにより、上位バスのスループットを向上させることができる。ここでスプリットとは、トランザクション発行のサイクルとそのトランザクションに対する応答のサイクルとを分割可能とし、その間にトランザクション発行元のプロセッサにバス使用権を放棄させ、他のバスマスタがトランザクションを発行できるようにする方式である。

【0010】プロセッサのトランザクションの要求データがブリッジキャッシュに存在しない場合、下位バスにトランザクションを転送してメインメモリや他のクラスタのキャッシュを読みに行く必要がある。この場合において、下位バスからの応答を待つ間、そのトランザクションを発したプロセッサが上位バスを占有していると、上位バスのスループットが低下してしまう。これに対して、本発明の構成では、プロセッサのトランザクションの要求データがブリッジキャッシュに存在せず下位バスにトランザクションを転送する必要がある場合には、そのトランザクションをスプリットすることができるので、その間上位バスに他のトランザクションを発行することができ、全体として上位バスのスループットを向上させることができる。

【0011】また、本発明は、さらに上位バス上のプロセッサが、それぞれ自分専用のプロセッサキャッシュを有する場合において、プロセッサキャッシュ及びブリッジキャッシュは各キャッシュブロックのステートを記憶

するステート記憶手段をそれぞれ有し、プロセッサ及びバスブリッジは自らが有するプロセッサキャッシュ又はブリッジキャッシュのステート記憶手段を制御するステート制御手段をそれぞれ有し、プロセッサキャッシュの各キャッシュブロックの取り得るステートを、ブリッジキャッシュの対応キャッシュブロックのステートに基づいて制約することにより、ブリッジキャッシュとこのブリッジキャッシュの上位のプロセッサキャッシュとのマルチレベル包含性を維持することを特徴とする。

【0012】この構成では、プロセッサキャッシュ及びブリッジキャッシュのそれぞれに各キャッシュブロックのステートを記憶するステート記憶手段を設け、このステート記憶手段に記憶されたステートを用いてトランザクションの制御を行う。ステート記憶手段における各ステートは、ステート制御手段によって制御する。ここで、この構成では、プロセッサキャッシュの各キャッシュブロックの取り得るステートをブリッジキャッシュの対応キャッシュブロックのステートに基づいて制約することにより、ブリッジキャッシュとこのブリッジキャッシュの上位のプロセッサキャッシュとのマルチレベル包含性を維持する。この結果、プロセッサキャッシュとブリッジキャッシュとメインメモリとのキャッシュコンシステンシを保つことができる。しかも、プロセッサキャッシュとブリッジキャッシュとの間でマルチレベル包含性を維持することにより、キャッシュコンシステンシ維持のためのバスブリッジの構成を比較的簡単なものとすることができる。

【0013】また、本発明は、バスブリッジに対してブリッジキャッシュのエビクション対象のキャッシュブロックのデータを格納するバッファ手段を設け、エビクション判定手段にてエビクションが必要と判定されると、ブリッジキャッシュのエビクション対象のキャッシュブロックのデータを前記バッファ手段に退避させたのちブリッジキャッシュの当該キャッシュブロックを用いて上位バスからのトランザクションに対する処理を実行し、前記バッファ手段に退避したデータを用いてライトバック処理を行うことを特徴とする。

【0014】この構成では、まずバスブリッジが上位バスからトランザクションを受け取った場合、エビクション判定手段によって当該トランザクションのアドレスに対応するブリッジキャッシュのキャッシュブロックをエビクションする必要があるか否かを判定する。この判定の結果エビクションが必要と判定された場合、バスブリッジは、ブリッジキャッシュの当該エビクション対象のキャッシュブロックをバッファ手段に書き込む。この結果、当該エビクション対象のキャッシュブロックのデータのライトバック処理はバッファ手段に書き込まれたデータによって行うことができるので、ブリッジキャッシュの当該エビクション対象のキャッシュブロックのデータを確保しておく必要がなくなる。したがって、当該キ

ャッシュブロックは上位バスからのトランザクションによって使用可能な状態となるので、バスブリッジは、ライトバック処理の終了を待たずに当該キャッシュブロックを用いて上位バストランザクションに対する処理を進行することができる。

【0015】このように、本構成によれば、トランザクションに対する処理をエビクションに先行して行うことができるため、プロセッサが最初にトランザクションを発行してからそのトランザクションが完結するまでの時間を大幅に短縮することができる。

【0016】なお、本構成において、さらにエビクション判定に応じてバスブリッジから上位バスに発行したインバリデートトランザクションに対して、上位バスのプロセッサから当該エビクション対象のキャッシュブロックのデータがライトバックされてきた場合に、プロセッサからライトバックされてきたデータによって前記バッファ手段に格納されているデータを更新する構成とすることもできる。この構成によれば、当該バスブリッジの上位のプロセッサに当該エビクション対象のキャッシュブロックの最新のデータが存在する場合に、この最新のデータはバスブリッジのバッファ手段に書き戻されるので、バスブリッジにおけるライトバックにおいては、その最新データをバッファ手段からメインメモリに書き戻すことができる。

【0017】また、本発明は、バスブリッジが、当該バスブリッジにおいてスプリット中のトランザクションの有無を示すスプリットフラグ手段と、上位バスから受け取ったトランザクションの種類がキャッシュブロックを使用する種類であるか否かを判定するトランザクション判定手段と、上位バスから受け取ったトランザクションのリトライの要否を判定するリトライ判定手段であって、前記スプリットフラグ手段がスプリット中のトランザクションが有ることを示し、かつ上位バスから受け取ったトランザクションが前記トランザクション判定手段によってキャッシュブロックを使用する種類であると判定された場合にリトライが必要と判定するリトライ判定手段と、前記リトライ判定手段によってリトライが必要と判定されたときに、上位バスに対して前記トランザクションについてのリトライ信号を出力するリトライ出力手段とを有し、前記上位バスのトランザクションを発行したプロセッサは、バスブリッジからリトライ信号を受け取った場合に、当該トランザクションをいったん終了し、所定時間後に前記トランザクションを再発行することを特徴とする。

【0018】この構成において、スプリット中とは、バスブリッジが上位バスからのトランザクションをスプリットしてから、当該トランザクションの発行元のプロセッサに対してスプリット応答が完了するまでの間のことをいう。この構成では、バスブリッジがあるトランザクションについてスプリット中の間に別のトランザクシ

ンを受け取った場合、当該別のトランザクションがキャッシュブロックを使用するかしないかを判定し、使用すると判定される場合にのみ当該別のトランザクションをリトライさせる。そして、キャッシュブロックを使用しない種類のトランザクションについては、リトライさせずにそのまま処理を行う。したがって、バスブリッジに起因するリトライの発生頻度の増大を防止し、システム全体としての処理効率を高めることができる。

【0019】また、本発明は、バスブリッジが、当該バスブリッジにおいてスプリット中のトランザクションが使用するキャッシュブロックのアドレス情報を記憶するスプリット情報記憶手段と、上位バスから受け取ったトランザクションのアドレスと前記スプリット情報記憶手段に記憶されたアドレス情報とを比較し、上位バスから受け取ったトランザクションがスプリット中のトランザクションの使用するか否かを判定するアドレス判定手段と、上位バスから受け取ったトランザクションのリトライの要否を判定するリトライ判定手段とを有し、前記アドレス判定手段において上位バスから受け取ったトランザクションがスプリット中のトランザクションの使用するか否かを判定すると判定された場合にリトライが必要と判定するリトライ判定手段とを有し、前記上位バスのトランザクションを発行したプロセッサは、バスブリッジからリトライ信号を受け取った場合に、当該トランザクションをいったん終了し、所定時間後に前記トランザクションを再発行することを特徴とする。

【0020】この構成では、スプリット情報記憶手段に格納されたアドレス情報を用いることにより、バスブリッジが受け取った上位バスからのトランザクションが現在スプリット中のトランザクションの使用するか否かを判定することができる。そして、この構成では、上位バスからのトランザクションが、現在スプリット中のトランザクションの使用するか否かを判定すると判定された場合にのみ、当該上位バスからのトランザクションのリトライを行う。したがって、この構成によれば、バスブリッジに起因するリトライの発生頻度が小さくなり、システム全体としての処理効率を高めることができる。

【0021】また、本発明は、前記バッファ手段を有する構成において、さらにバスブリッジが、上位バスから受け取ったトランザクションのアドレスとバッファ手段に退避中のキャッシュブロックのアドレスとを比較し、当該トランザクションが退避中のキャッシュブロックに対するトランザクションであるか否かを判定するアドレス比較手段と、前記上位バスからのトランザクションが退避中のキャッシュブロックに対するトランザクションだと判定された場合に、当該トランザクションに対してリトライが必要と判定するリトライ判定手段と、前記リトライ判定手段によってリトライが必要と判定されたと

きに、上位バスに対して前記トランザクションについてのリトライ信号を出力するリトライ出力手段とを有し、前記上位バスのトランザクションを発行したプロセッサは、バスブリッジからリトライ信号を受け取った場合に、当該トランザクションをいったん終了し、所定時間後に前記トランザクションを再発行することを特徴とする。

【0022】この構成は、エビクション対象のキャッシュブロックをブリッジキャッシュから一時的に退避しておくためのバッファ手段を有するバスブリッジを用いるシステムに関するものである。バッファ手段に退避しているデータは、データの整合性を保つため他のトランザクションからは読み書きができない。したがって、バスブリッジが当該退避データに対する別のトランザクションを受け取った場合、当該トランザクションについてリトライを行う必要がある。そこで、本構成では、アドレス比較手段によって、上位バスから受け取ったトランザクションのアドレスとバッファ手段に退避中のデータのアドレスとを比較し、退避中のデータに対するトランザクションのみをリトライさせる。この構成によれば、バッファ手段に退避中のデータに対するトランザクションのみをリトライさせることができる。

【0023】また、本発明は、バスブリッジが、上位バスから受け取ったトランザクションをスプリットして下位バスにトランザクションを発行した場合において、当該下位バスのトランザクションに対して下位バスからリトライ信号を受け取った場合には、前記上位バスに対してスプリット応答トランザクションを発行すると共に前記上位バスのトランザクション発行元のプロセッサに対してリトライ信号を出力し、前記トランザクション発行元のプロセッサは、前記バスブリッジからのリトライ信号を受信すると、前記トランザクションをいったん終了し、所定時間後にリトライすることを特徴とする。

【0024】この構成は、バスブリッジが、上位バスから受け取ったトランザクションに対応して下位バスにトランザクションを発行した場合において、当該下位バス・トランザクションに対して下位バスのエージェント（例えばI/O装置）などからリトライ信号がバスブリッジに返ってきた場合の処理に関する。すなわち、この構成では、バスブリッジ自身が当該下位バス・トランザクションについてのリトライ処理を行うのではなく、バスブリッジからトランザクション発行元のプロセッサに対してリトライ信号を発行し、そのプロセッサにリトライ処理をさせる。この構成によれば、バスブリッジがリトライ処理のための特別の構成を持つ必要がなくなり、バスブリッジの構成を単純化することができる。

【0025】また、本発明は、バスブリッジが、上位バス・トランザクションが有するスプリット識別子に対して下位バスにおいて当該バスブリッジに固有のブリッジ識別子を付加することにより、下位バス・トランザクション

のスプリット識別子を生成するスプリット識別子拡張手段を有し、上位バスから受け取った上位バストランザクションをスプリットして下位バスに下位バストランザクションを発行する場合に、前記スプリット識別子拡張手段によって得られたスプリット識別子を下位バストランザクションに付することを特徴とする。

【0026】この構成において、上位バストランザクションのスプリット識別子は、トランザクションの発行元のプロセッサを識別するための識別子と当該プロセッサ内においてトランザクションを識別するための識別子とを含んでいるが、スプリット識別子のビット数が定まっていれば識別可能なプロセッサの数、すなわちシステムに接続可能なプロセッサの数も自ずと制限される。この接続可能プロセッサ数の論理的制限を緩和するのが本構成である。

【0027】この構成では、バスブリッジにて上位バストランザクションをスプリットして下位バスにトランザクションを発行する場合には、上位バストランザクションに与えられたスプリット識別子に対してバスブリッジのブリッジ識別子を付加することにより拡張したスプリット識別子を下位バストランザクションに与える。この構成によれば、下位バストランザクションは、拡張されたスプリット識別子に含まれるブリッジ識別子よりその発行元のバスブリッジを特定することが可能となる。したがって、下位バストランザクションからバスブリッジが特定されれば、トランザクション発行元プロセッサはそのバスブリッジの上位にあることになるので、別々の上位バス上のプロセッサに同一のプロセッサ識別子を与えたとしてもそれらを互いに識別することが可能となる。したがって、この構成によれば、プロセッサ識別子の数に制限があっても、ブリッジ識別子を加えたことにより、システム全体として識別可能なプロセッサの数を増やすことができ、階層バスシステム全体に接続可能なプロセッサ数を増やすことができる。

【0028】また、本発明は、バスブリッジが、一方のバスから受け取ったトランザクションのスプリット識別子を所定ビット数のブリッジトランザクション識別子に変換し、他方のバスにおいて当該バスブリッジに固有なブリッジ識別子に前記ブリッジトランザクション識別子を付加することにより、他方のバスに発行するトランザクションに付与するスプリット識別子を生成するスプリット識別子生成手段を有し、前記ブリッジトランザクション識別子のビット数と前記ブリッジ識別子のビット数との和は前記一方のバスにおけるスプリット識別子のビット数に等しく、一方のバスから受け取ったトランザクションに対し他方のバスにトランザクションを発行する必要がある場合には、当該他方のバスに発行するトランザクションに対して前記スプリット識別子生成手段によって得られたスプリット識別子を当該他方のバスに発行するトランザクションに付することを特徴とする。

【0029】この構成では、バスブリッジが一方のバスから受け取ったトランザクションも対応して他方のバスにトランザクションを発行する際に当該他方のバスのトランザクションに与えるスプリット識別子を、前記一方のバスのトランザクションのスプリット識別子と同ビット数とする。ここで、バスブリッジがトランザクションに対して与えるスプリット識別子には、当該バスブリッジのブリッジ識別子を含める。このため、バスブリッジは、一方のバスからのトランザクションのスプリット識別子を所定ビット数のブリッジトランザクション識別子に変換し、このブリッジトランザクション識別子と当該バスブリッジのブリッジ識別子とを組み合わせることにより、他方のバスへ発行するトランザクションのスプリット識別子を生成する。ここで、ブリッジトランザクション識別子のビット数は、スプリット識別子のビット数からブリッジ識別子のビット数を減算した値である。なお、この構成においては、一つのバスブリッジに対して上位バスに対するブリッジ識別子と下位バスに対するブリッジ識別子が与えられ、各バスに対するブリッジ識別子としては、当該バスにおいて当該バスブリッジを一意的に識別できる値が与えられる。このような構成によれば、下位バスにもプロセッサを接続することが可能になる。

【0030】なお、この構成において、さらにスプリット識別子プロセッサ識別子とブリッジ識別子を同ビット数にすれば、プロセッサとバスブリッジとを識別子に関して等価として扱うことができるので、下位バスにプロセッサを接続したり、上位バスにバスブリッジを接続したりすることができ、システム構成の柔軟性を向上させることができる。また、さらに、バスブリッジが付与可能なブリッジトランザクション識別子が現在使用されているか否かを管理する管理テーブルを用意し、この管理テーブルを用いてブリッジトランザクション識別子を決定することにより、ブリッジトランザクション識別子を無駄なく割り当てることができる。

【0031】また、本発明は、階層バスシステムにおいてバスの物理的仕様及び論理的仕様を下位バスと上位バスとで共通としたことを特徴とする。

【0032】この構成によれば、上位バスと下位バスとの仕様を同一にすることにより、プロセッサ及びバスブリッジを上位バス及び下位バスのいずれにも接続することができるようになるので、システム構成のバリエーションを広げることができる。

【0033】また、本発明は、階層バスシステムにおいて、下位バスを複数設け、これら各下位バスにそれぞれメインメモリを接続し、各上位バスに対して下位バスと同数のバスブリッジを接続し、前記各上位バスと各下位バスとを互いに1つのバスブリッジを介して接続したことを特徴とする。

【0034】この構成では、1つのプロセッサは、複数

のバスブリッジを介して複数のメインメモリに接続される。各メインメモリはそれぞれ別々の下位バスに接続されているので、複数のメインメモリの総メモリサイズと同じメモリサイズの一つのメインメモリを採用するシステム構成をとった場合に比べて、下位バスの付加を低減することができ、下位バスのスループットを向上させることができる。なお、この構成において、複数の下位バスのそれぞれに対してバスブリッジを設け、これらバスブリッジを介して前記各下位バスを1つのI/Oバスに接続し、このI/OバスにI/O装置を接続すれば、必要なI/O装置を各下位バスごとに設ける必要がなくなり、システムの大規模化を防止することができる。

【0035】

【発明の実施の形態】以下、本発明に係るマルチプロセッサシステムの好適な実施形態を図面に基づいて説明する。

【0036】実施形態1。図1は、本発明に係るマルチプロセッサシステムの全体的な構成の一例を示す概略構成図である。

【0037】図1において、CPU10～15は、2つのグループ(CPU10～12、CPU13～15)にグループ分けされ、各グループごとに1本の上位バス20、22に接続されている。

【0038】各上位バス20、22は、それぞれバスブリッジ30、32に接続されている。各バスブリッジ30、32は、それぞれキャッシュ(以下、ブリッジキャッシュと呼ぶ)31、33を有している。ブリッジキャッシュ31は、上位バス20上のCPU10～12によって共有され、ブリッジキャッシュ33は、上位バス22上のCPU13～15によって共有される。このように、複数のCPUをグループ化して1つのバスブリッジ(及びブリッジキャッシュ)に接続したものをクラスと呼ぶ。各バスブリッジ30、32は、下位バス40によって相互接続され、この下位バス40にメインメモリ50及びI/O装置52が接続される。なお、I/O装置52は1つとは限らず、複数接続することもできる。

【0039】したがって、例えば、CPU11から上位バス20に発行されたトランザクションがリード(Read)であった場合には、バスブリッジ30は、ブリッジキャッシュ31をスヌープ(検索)し、この結果ブリッジキャッシュ31にヒットした場合は検索されたデータを上位バス20に返し、ヒットしなかった場合には下位バス40にリード・トランザクションを発行する。

【0040】図1の構成において、上位バス20及び22はスプリットバスであり、CPUから発せられるメモリトランザクション(メインメモリのデータ内容についてのトランザクション)及びI/Oトランザクション(I/O装置に対するトランザクション)の両方についてスプリット可能となっている。また、CPU10～15は、スプリット対応可能なCPUである。トランザク

ションをスプリットをするかしないかの判断はバスブリッジ30及び32で行われる。

【0041】ここで、トランザクションをスプリットした場合におけるシステム各部の動作について説明する。上位バスに発せられたトランザクションをスプリットすると決定した場合、バスブリッジは、トランザクション発行元のCPUに対して、当該トランザクションをスプリットする旨の応答(「スプリット終了」と呼ぶ)を返すとともに、下位バスに対して必要なトランザクションを発行する。一方、スプリット終了信号を受け取った要求元CPUは、いったん上位バスを解放し、下位バスからの応答を待つ。バスブリッジは、下位バスに発行したトランザクションに対する他のバスブリッジやメインメモリなどからの応答が揃ったところで、トランザクション発行元CPUに対するスプリット応答トランザクションを生成し、上位バス上に発行する。そして、トランザクション発行元CPUがこのスプリット応答トランザクションを受け取ったところで、当該CPUが最初に発行したトランザクションに対する一連の処理が終了する。このように、本構成によれば、上位バスをスプリットバスとしたことにより、トランザクションをスプリットしてからスプリット応答トランザクションが発行されるまでの間は上位バスが解放されているので、その間上位バスに他のトランザクションを発行することができる。

【0042】このように、図1の構成によれば、まずCPU10～15とメインメモリ50、I/O装置52との間が階層的なバス構成となっているため、個々の上位バス20、22に接続できるCPUの数が少なくても、システム全体として多数のCPUを接続することができる。

【0043】また、この構成では、ブリッジキャッシュ31、33を設けたことにより、上位バス20、22のトランザクションの要求データがブリッジキャッシュに存在する場合には、メインメモリ50を読みに行く必要がなくなるため、下位バス40のバス負荷が低減され、この結果下位バス40のスループットが向上する。また、CPUが自らの発したトランザクションに対するレスポンスを得るまでのレイテンシも短くなる。

【0044】さらに、この構成では、上位バス20、22をスプリットバスとしたことにより、上位バスのスループットを向上させることができる。すなわち、この構成では、上位バス20、22のトランザクションを下位バス40に転送する必要がある場合には、そのトランザクションをスプリットすることができるので、その間上位バス20、22に他のトランザクションを発行することができ、全体として上位バスのスループットを向上させることができる。

【0045】なお、図1の構成はあくまで一例であり、上位バス及び下位バスには、物理的制約を満たす範囲内であれば、CPU及びバスブリッジをいくつでも接続す

ることができる。

【0046】さて、図1の構成において、CPU10～15にそれぞれ専用キャッシュ（以下、CPUキャッシュと呼ぶ）を設けた場合を考える。このように各CPUに専用のCPUキャッシュを設ければ、上位バス20、22に発行されるトランザクションが少なくなるので、上位バス20、22のスループットを向上させることができる。ただし、この場合には、CPUキャッシュ、ブリッジキャッシュ、及びメインメモリの間のデータの貫性（キャッシュ・コンシステンシ）を保つ必要がある。

【0047】キャッシュ・コンシステンシを維持するためのプロトコルとしては、従来より様々な方式が提案されているが、その代表的なものにMESIプロトコルがある。MESIプロトコルは、ライトバック・キャッシュのためのキャッシュ・コンシステンシ・プロトコルの一つであり、各キャッシュブロックに対してM、E、S、Iの4種類のいずれかのステート（状態）を与え、キャッシュブロックの読み込み、書き換えの際にそのステートを一定の規則にしたがって制御することによりキャッシュ・コンシステンシを維持する。

【0048】ここで、M、E、S、Iの各ステートの示す意味は以下の通りである。

【0049】M(Modify)：そのブロックの更新されたデータ（すなわち、そのブロックの最新のデータ）をそのキャッシュだけが持つ。

【0050】E(Exclusive)：そのブロックに関しメインメモリと同一内容のデータをそのキャッシュだけが持つ。

【0051】S(Shared)：そのブロックに関しメインメモリと同一内容のデータを持つが、他のキャッシュも当該データを持っている可能性がある。

【0052】I(Invalid)：そのブロックは無効である。

【0053】キャッシュブロックの読み込み、書き換えなどの操作を行った場合のステートの制御については、MESIプロトコルに規則が定められている。例えば、あるCPUのキャッシュのステートSのキャッシュブロックにライト（書込み）が行われる場合について説明する。この場合、他のCPUのキャッシュが当該キャッシュブロックをステートSで持っている可能性がある。したがって、ライトを行うCPUは、他のキャッシュの当該キャッシュブロックを無効化するため、インバリデート(Invalidate：無効化)・トランザクションを発行する。これを受けた他のCPUは、自らのキャッシュの当該キャッシュブロックのステートをIとし、ライトを行うCPU自体は所望のライト処理を行ったのち当該キャッシュブロックのステートをMに変更する。このような操作により、CPUがキャッシュ内の古いデータを使用するおそれが無くなり、キャッシュ・コンシステンシが

保たれる。すなわち、あるキャッシュがステートSのブロックを持っている場合において当該ブロックについてライト（データの書換え）を行う場合には、キャッシュ・コンシステンシ維持のためには、他のキャッシュの当該ブロックを無効化する。

【0054】また、別の例としては、あるキャッシュがステートEのブロックを持っている場合において当該ブロックに対してライトを行う場合には、他のキャッシュはすべて当該ブロックを持っていない（ステートI）ので、他のキャッシュに対して何ら手当てをすることなくライト処理を行うことができる。MESIプロトコルには、その他詳細な規則が定められているが、ここでは省略する。

【0055】本実施形態では、基本的にこのMESIプロトコルを用いてキャッシュ・コンシステンシの管理を行うが、MESIプロトコル自体は1階層のキャッシュシステムについて規定したものであり、本実施形態のようなキャッシュが複数階層構成となっているものに対しては、そのまま利用することはできない。そこで本実施形態では、上記MESIプロトコルを次のように拡張する。

【0056】すなわち、この拡張プロトコルでは、MESIの各ステートは、同レベルにあるキャッシュ同士の関係を示すものと規定する。CPUキャッシュのステートは他のCPUキャッシュとの関係を示すものとし、ブリッジキャッシュのステートは他のブリッジキャッシュとの関係を示すものとする。この場合、CPUキャッシュのステートは前述したものと同じ意味となるが、ブリッジキャッシュのステートの意味は以下に示す通りとなる。

【0057】M：そのブロックの更新されたデータ（すなわち、そのブロックの最新のデータ）を、そのブリッジキャッシュ自体又はそのブリッジキャッシュの上位のCPUキャッシュが持つ。

【0058】E：そのブリッジキャッシュは、そのブロックに関しメインメモリと同一内容のデータを有しており、他のブリッジキャッシュは当該ブロックのデータを有しない。

【0059】S：そのブロックに関しメインメモリと同一内容のデータを持つが、他のブリッジキャッシュも当該データを持っている可能性がある。

【0060】I：そのブロックは無効である。

【0061】なお、ここで、「ブリッジキャッシュの上位のCPUキャッシュ」とは、あるブリッジキャッシュに対し、バスブリッジ及び上位バス経由で接続されたCPUのキャッシュのことを示すものとする。逆に「CPUキャッシュの下位のブリッジキャッシュ」とは、あるCPUキャッシュに対し、上位バス及びバスブリッジを経由して接続されたブリッジキャッシュのことを示すものとする。また、「キャッシュのステート」と言った場

合、そのキャッシュにおける、現在注目しているキャッシュブロック（すなわち、現在のトランザクションの要求アドレスに対応するキャッシュブロック）のステータスのことを示すものとする。

【0062】そして、本実施形態では、ブリッジキャッシュのステータスとCPUキャッシュのステータスとの関係に一定の制約を与えることにより、ブリッジキャッシュとCPUキャッシュとのマルチレベル包含性（MLI）を保証し、多階層のキャッシュシステムにおけるキャッシュ・コンシステンシを維持する。ここで、MLIとは、多階層のキャッシュシステムにおけるキャッシュ・コンシステンシの制御を容易ならしめるために採用される制約の一種であり、CPUキャッシュに存在するデータは必ずその下位のブリッジキャッシュに存在するという性質のことをいう。なお、本実施形態においては、MLIを採用するため、ブリッジキャッシュの容量は、少なくともその上位のCPUキャッシュの容量の総和以上となっている。MLIを維持するようにキャッシュを制御するようにすれば、システムを簡単な回路で構成することができ、余計なバス・トランザクションの発生を抑

制することができる。

【0063】本実施形態では、MLIを維持するために、ブリッジキャッシュのステータスに基づきCPUキャッシュの取り得るステータスに制約を与える。本実施形態では、この制約として、以下に示す2種類の制約のいずれかを採用する。

【0064】まず、第1の制約（以下、「制約1」と呼ぶ）を表1に示す。

【0065】

【表1】

ブリッジキャッシュ	CPUキャッシュ
M	M, E, S, I
E	S, I
S	S, I
I	I

次に、第2の制約（以下、「制約2」と呼ぶ）を表2に示す。

【0066】

【表2】

ブリッジキャッシュ	CPUキャッシュ
M	M, E, S, I
E	M, E, S, I
S	S, I
I	I

表1及び表2は、ブリッジキャッシュのステータスがM, E, S, Iのそれぞれの場合におけるそのブリッジキャッシュの上位のCPUキャッシュが取り得るステータスの種類を示している。

【0067】制約1及び制約2は、ブリッジキャッシュのステータスがM, S, Iの時には共通である。

【0068】制約1及び制約2では、ブリッジキャッシュのステータスがIである場合にCPUキャッシュのステータスはIしか認めないことにより、ブリッジキャッシュに無いデータ（キャッシュブロック）はその上位のリーダーCPUキャッシュにも無いということを保証する。

【0069】また、制約1及び制約2では、ブリッジキャッシュのステータスがSである場合には、CPUキャッシュのステータスにS又はIを認める。前述したように、ブリッジキャッシュのステータスがSとなるのは、ブリッジキャッシュが下位バスから新たなデータを読み込んだときに他のブリッジキャッシュが同一データをもっている場合である。ここで、CPUキャッシュのステータスを制約1又は2のごとく制限しておけば、CPUキャッシュがそのデータをブリッジキャッシュから読み込んだときにCPUキャッシュのステータスがSとなる。これにより、次にそのCPUキャッシュにおいてそのデータの書換え（ライト）が行われる場合には、必ず他のキャッシュの無効化のためのトランザクション（インバリデー

ト：Invalidate）が発行される。そして、そのインバリデートをそのCPUの下位のバスブリッジが受け取ってそのバスブリッジから下位バスにもインバリデートを発行することにより、他のブリッジキャッシュ（及びその上位のCPUキャッシュ）にある当該データを無効化することができる。

【0070】また、制約1及び制約2では、ブリッジキャッシュのステータスがMである場合にCPUキャッシュのステータスとしてM, E, S, Iのすべてのステータスを認める。本実施形態において、ブリッジキャッシュのステータスがMになるのは、上位のCPUにおいてライト動作が行われた場合である。したがって、この場合は、CPUキャッシュにはすべてのステータスを認めておく必要がある。

【0071】制約1と制約2の相違点は、ブリッジキャッシュがEのときに制約2ではCPUキャッシュのステータスとしてM, Eを認めるが、制約1ではM, Eを認めないという点である。

【0072】ブリッジキャッシュがEとなるのは、ブリッジキャッシュが下位バスから新たなデータを読み込んだときに他のブリッジキャッシュが同一データをもっていない場合である。この場合において、あるCPUキャッシュがブリッジキャッシュからそのデータを読み込んだ場合、制約2では同一上位バス上の他のCPUキャッ

シュがそのデータを持っていない場合には、そのCPUキャッシュのステートをEとすることを認める。これに対して制約1では、同じ状況でそのCPUキャッシュのステートをSとする。

【0073】制約2では、CPUキャッシュがあるデータをステートEで持っている場合においてそのデータを書き替えるときには、そのCPUは外部に何ら信号を発せず書き替えを実行し、その結果CPUキャッシュのステートがMに変わる。この制約2では、ブリッジキャッシュのステートがEである場合に、その上位のCPUが最新のデータを持っているかどうかを知るには、上位バスをスヌープするなどの処理が必要がある。

【0074】一方、制約1では、ブリッジキャッシュがあるデータをステートEで持っている場合に、その上位のCPUキャッシュが当該データを持っている場合にはそのステートはSになる。したがって、CPUキャッシュが当該データを書き替える場合には上位バス上に必ずインバリデートが発行されるため、ブリッジキャッシュはそのインバリデートを受け取ることにより、CPUキャッシュのデータが書き替えられたことを知ることができる。このときブリッジキャッシュのステートはMに変わる。したがって、制約2によれば、ブリッジキャッシュのステートがEのときには、その上位のCPUキャッシュには最新のデータが存在しないことが保証される。

【0075】本実施形態では、このように、ブリッジキャッシュとその上位のCPUキャッシュとのステートの関係に制約を与えることにより、無用なバス・トランザクションを増やさずにキャッシュ・コンシステンシを維持することができる。

【0076】[バスブリッジの動作] 本実施形態では、バスブリッジの働きにより、以上説明した制約を満たし、キャッシュ・コンシステンシを維持する。以下、本実施形態のバスブリッジ及びブリッジキャッシュの動作について説明する。

【0077】[1] 上位バスに発行されたトランザクションに対する基本動作

まず、CPUから上位バスに発行されたトランザクションを受け取った場合のバスブリッジの一連の動作について説明する。

【0078】本実施形態においては、上位バスには以下のトランザクションが定義されている。

【0079】(1) Read (リード) : アドレスで示されたキャッシュブロックを読み込む。CPUからリードのリクエストが出され、CPUキャッシュでミスした場合(すなわち、リード・リクエストの要求データがCPUキャッシュ内になかった場合)に発行される。

【0080】(2) Invalidate (インバリデート) : 他のCPUキャッシュに存在するアドレスで示されたキャッシュブロックを無効化する。CPUからラ

イトのリクエストが出され、CPUキャッシュでヒットしたもののそのCPUキャッシュのステートがSであった場合(ライト・リクエストの要求データがCPUキャッシュ内にステートSで存在した場合)に発行される。

【0081】(3) Read&Invalidate (リード・アンド・インバリデート) : 他のCPUキャッシュに存在するアドレスで示されたキャッシュブロックを無効化し、当該アドレスで示されたキャッシュブロックを読み込む。CPUからライト・リクエストが出され、CPUキャッシュでミスした場合に発行される。

【0082】(4) WriteBack (ライトバック) : アドレスで示されたキャッシュブロックを下位のブリッジキャッシュに書き戻す。CPUのリクエストによりCPUキャッシュに空きブロックを作る必要があるときに発行される。

【0083】なお、以上は、メインメモリのデータに対するトランザクション(メモリ・トランザクション)のみである。このほかにもI/O装置に対するトランザクション(I/Oトランザクション)があるが、これについての説明は省略する。

【0084】[1. 1] 下位バス出力判定
上位バスにトランザクションが発行された場合、バスブリッジはまずそのトランザクションをスプリットする必要があるか判定し、スプリットする場合には下位バスに発行するトランザクションの種類を決定する。

【0085】ここで、下位バスには、次に示すトランザクションが定義されている。

【0086】(1) Read (リード) : 指定されたアドレスのキャッシュブロックを読み込む。

【0087】(2) Invalidate (インバリデート) : 他のブリッジキャッシュに存在する指定されたアドレスのキャッシュブロックを無効化する。

【0088】(3) Read&Invalidate (リード・アンド・インバリデート) : 他のブリッジキャッシュに存在する指定されたアドレスのキャッシュブロックを無効化し、当該アドレスで示されたキャッシュブロックを読み込む。

【0089】(4) WriteBack (ライトバック) : 指定されたアドレスのキャッシュブロックを下位のメインメモリに書き戻す。

【0090】なお、下位バスにも上位バス同様I/Oトランザクションが定義されているが、ここでは省略する。

【0091】[1. 1. 1] 制約1を採用する場合
制約1を採用する場合には、バスブリッジは表3に示す規則にしたがってスプリットの要否の判定及び下位バスに発行するトランザクションを決定する。

【0092】

【表3】

	上位バス・トランザクション			
ブリッジキャッシュ ステート	Read	Read & Invalidate	Invalidate	Write Back
M	○	○	○	○
E	○	○	○	×
S	○	Invalidate	Invalidate	×
I	Read	Read&Invalidate	×	×

○：下位バスにトランザクションを発行しない

×：あり得ない

表3に示すように、バスブリッジは、自らの上位バスに発行されたトランザクションの種類とそのトランザクションを受け取ったときのブリッジキャッシュのステートとに基づき、スプリットするか否か及びスプリットする場合における下位バスに発行するトランザクションの種類を決定する。なお、以下においては、「ブリッジキャッシュがX（XはM、E、S、Iのうちのどれかのステート）である」とは、バスブリッジがトランザクションを受け取った場合において、そのトランザクションのアドレスに対応する当該ブリッジキャッシュのブロックのステートがXであることを意味する。

【0093】表3に示すように、上位バスにReadが発行された場合、バスブリッジは、ブリッジキャッシュがIの場合にのみ、上位バスのReadをスプリットして下位バスにReadを発行する。なお、ブリッジキャッシュがM、E、Sのいずれかである場合は、Readトランザクションの要求アドレスはブリッジキャッシュ又は同一上位バス上のCPUキャッシュに存在するので、要求元CPUはそれらから要求データを得ることができる。したがって、この場合は下位バスにはトランザクションが発行されない。なお、バスブリッジが上位バスのReadをスプリットしなかった場合には、上位バスだけでトランザクションが完了し、ブリッジキャッシュのステートはトランザクションの前と変化しない。

【0094】上位バスにRead&Invalidateが発行された場合は、バスブリッジは、ブリッジキャッシュがS又はIのときにトランザクションをスプリットする。まず、ブリッジキャッシュがSの場合には、バスブリッジは上位バスのRead&Invalidateをスプリットし、下位バスに対してInvalidateを発行する。この場合は、Read&Invalidateで要求されるデータはブリッジキャッシュに存在するので、バスブリッジは、当該データをスプリット

終了信号と共に要求元CPUに返し、他のバスブリッジのブリッジキャッシュを無効化するために下位バスにInvalidateを発行する。一方、ブリッジキャッシュのIの場合には、バスブリッジはトランザクションをスプリットし、下位バスに対してRead&Invalidateを発行する。この場合は、Read&Invalidateで要求されるデータはブリッジキャッシュに存在しないので、バスブリッジは、当該データを入手しかつ他のバスブリッジのブリッジキャッシュを無効化するため、下位バスにRead&Invalidateを発行する。なお、ブリッジキャッシュがM又はEの場合は、バスブリッジはトランザクションをスプリットしない。ステートがM又はEということは、当該ブリッジキャッシュ又はその上位のCPUキャッシュに要求データが存在し、他のブリッジキャッシュ及びその上位のCPUキャッシュにはその要求データが存在しないからである。なお、ブリッジキャッシュがM又はEの場合は、トランザクション終了後のブリッジキャッシュのステートは共にMとなる。

【0095】上位バスにInvalidateが発行された場合は、バスブリッジは、ブリッジキャッシュがSの場合に、上位バスのInvalidateをスプリットし、下位バスに対してInvalidateを発行する。ブリッジキャッシュがSであるため、他のブリッジキャッシュが同じデータを持っている可能性があるからである。そして、ブリッジキャッシュのステートがM又はEの場合は、バスブリッジはトランザクションをスプリットしない。この場合、他のブリッジキャッシュは無効化対象のデータ（ブロック）を有していないからである。なお、ブリッジキャッシュがIである場合は、CPUから上位バス上にInvalidateが発行されることはない。ブリッジキャッシュがIの場合その上位のCPUキャッシュのステートは必ずIであるが、Inv

validateはCPUキャッシュがステートSのときにライト動作を行う場合にしか発行されないからである。なお、上位バスに発行されたInvalidateが終了した後は、ブリッジキャッシュのステートはMとなる。

【0096】また、上位バスに発行されたトランザクションがWriteBackであった場合は、トランザクションのスプリットは行われない。WriteBackトランザクションでは、CPUキャッシュにある最新データをブリッジキャッシュに書き戻すだけなので、他のバスブリッジには何ら影響がないからである。なお、WriteBackが発行されるのはCPUキャッシュがMのブロックを持っている場合だけであり、制約1ではCPUキャッシュがMとなり得るのは下位のブリッジキャッシュがMの場合だけである。したがって、ブリッジキャッシュがE、S、Iのときは上位バス上にWriteBackが発行されることはない。

【0097】以上説明した規則にしたがってバスブリッジが下位バスに発行するトランザクションを決定することにより、表1に示したブリッジキャッシュとCPUキャッシュとの制約1を満足させ、キャッシュ・コンシステンスを維持することができる。

【0098】そして、バスブリッジは、このようにして決定されたトランザクションを下位バスに発行する。この下位バスに発行されたトランザクションを受け取った他のバスブリッジの動作については後の〔2〕節において説明する。

【0099】次に、以上説明した規則の変形例について説明する。変形例では、上位バスのトランザクションの要求アドレスが、命令コードを指すものか、それともいわゆる「データ」（非命令コード）を指すものかを判定し、この判定を下位バスに発行するトランザクションの種類の決定に反映させる。ここで、Invalidate及びRead&Invalidate（以下、Invalidate系と呼ぶ）は、CPUがライトを行う場合に発行されるものなので、命令コードを要求することはほとんどない。したがって、この変形例では、Readについてのみ、命令コードを要求する場合とデータを要求する場合とでスプリット時に下位バスに発行するトランザクションの種類を変える。この変形例における規則を表4に示す。

【0100】

【表4】

ブリッジキャッシュ ステート	上位バス・トランザクション				
	Read		Read &Invalidate	Invalidate	WriteBack
命令コード	データ				
M	○	○	○	○	○
E	○	○	○	○	×
S	○	○	Invalidate	Invalidate	×
I	Read	Read &Invalidate	Read&Invalidate	×	×

○：下位バスにトランザクションを発行しない

×：あり得ない

表4は、Readの欄以外は表3と全く同じである。表4から分かるように、この変形例では、上位バスに発行されたReadをスプリットするときに、そのReadトランザクションが命令コードを要求するものである場合には下位バスにReadを発行し、データを要求するものである場合には下位バスにRead&Invalidateを発行する。

【0101】すなわち、この変形例では、上位バスのReadがデータ（非命令コード）を要求する場合は、スプリットの際にバスブリッジは、下位バスにRead&Invalidateを発行することにより、当該デー

タを得ると共に他のブリッジキャッシュの当該データのブロックを無効化する。これにより、バスブリッジは、下位バスから得られた当該データをブリッジキャッシュにステートMで登録することが可能となる。なお、当該データを読み込んだときにブリッジキャッシュのステートをMにするための機構については後のブリッジキャッシュの次ステートの決定方法の説明において説明する。

【0102】命令コードは、読み込まれた後に書き替えられることはないが、データ（非命令コード）は、読み込まれた後に書き替えられる可能性が高い。キャッシュ内のデータを書き替える場合、そのデータが含まれるキ

キャッシュブロックのステートがSであれば、Invalidateを発行して他のキャッシュの当該ブロックを無効化する必要がある。したがって、データをブリッジキャッシュに読み込んだときにステートSで登録したとすると、そのデータが書き替えられるときにバスブリッジから下位バスにInvalidateを発行する必要がある。この場合、データの読み込みのときに1回(Read)、データの書き替えのときに1回(Invalidate)の合計2回、下位バスにトランザクションが発行されることになる。これに対して、本変形例では、書き替えられる可能性が高いデータをブリッジキャッシュに読み込む必要があるときには、下位バスにReadではなくRead&Invalidateを発行することにより予め他のブリッジキャッシュを無効化し、そのデータをステートMでブリッジキャッシュに登録することにより、データの書き替えのときには下位バスにトランザクションを発行する必要がなくなる。したがって、本変形例では、データを読み込んだあとそのデータに書き替えを行う場合、読み込みの時点で下位バス

にRead&Invalidateを1回発行するのみでよく、上記表3の場合に比べて下位バスに発行するトランザクションの回数を減らすことができる。データ(非命令コード)は書き替えられる可能性は高いので、本変形例の手法による下位バス負荷低減の効果は高い。

【0103】なお、本変形例では、上位バスのReadが命令コードを要求するものであった場合は、表3の場合と同様、スプリット時には下位バスにReadを発行する。これにより、下位バスから得られた命令コードはバスブリッジにステートE又はSで登録される。

【0104】[1. 1. 2] 制約2を採用する場合
制約2を採用する場合、バスブリッジは、上位バスに発行されたトランザクションを受け取ると、そのトランザクションの種類とそのトランザクションを受け取ったときのブリッジキャッシュのステートとに基づき、表5に示す規則にしたがってスプリットの要否の判定及び下位バスに発行するトランザクションを決定する。

【0105】

【表5】

ブリッジキャッシュ ステート	上位バス・トランザクション			
	Read	Read & Invalidate	Invalidate	WriteBack
M	○	○	○	○
E	○	○	○	○
S	○	Invalidate	Invalidate	×
I	Read	Read&Invalidate	×	×

○：下位バスにトランザクションを発行しない

×：あり得ない

表5の規則と前記制約1に対する表3の規則との違いは、ブリッジキャッシュのステートがEのときに上位バスにWriteBackが発行される可能性があるか否かの違いである。制約1では、ブリッジキャッシュのステートがEの場合、CPUキャッシュにステートEを認めず、したがってCPUキャッシュがステートMとなることはない(表1参照)ので、CPUからWriteBackが行われることはない。これに対して、制約2では、CPUキャッシュにステートEを認めており(表2参照)、CPUキャッシュのステートEのブロックがCPUからの書き換えにより、バスブリッジが関知しないままステートMとなっている可能性がある。したがって、制約2では、ブリッジキャッシュがステートEのときに、上位のCPUキャッシュからWriteBackが行われる可能性がある。

【0106】表5の規則と前記表3の規則との相違はこの点のみであり、上位バス・トランザクションのスプリットの決定及び下位バスに発行するトランザクションの

決め方については、表5の規則は表3の規則と同一である。

【0107】以上説明した規則にしたがってバスブリッジが下位バスに発行するトランザクションを決定することにより、表2に示したブリッジキャッシュとCPUキャッシュとの制約2を満足させ、キャッシュ・コンシステンスを維持することができる。

【0108】[1. 2] ブリッジキャッシュの次ステート

次に、上位バスからのトランザクションによるブリッジキャッシュのステート遷移について、その上位バス・トランザクションをスプリットしない場合とスプリットする場合に分けて説明する。なお、ブリッジキャッシュの次ステートの決定の仕方は、制約1及び制約2のいずれを採用した場合も同じである。

【0109】まず、スプリットしない場合について説明

する。ここで、上位バス・トランザクションがRead又はWriteBackであった場合には、ブリッジキャッシュのステートは当該トランザクションの前後で変化しない。一方、上位バス・トランザクションがInvalidate系であった場合には、当該トランザクション終了後のブリッジキャッシュのステートはMとする。Invalidate系トランザクションはデータの書き替えのために発行されるからである。

【0110】次に、スプリットする場合について説明す

下位バスに発行した トランザクション	下位バスのスヌープ結果		
	ミス	HIT	HITM
Read	E	S	S
Invalidate	M	M	M
Read&Invalidate	M	M	M

ここで、スヌープ結果について説明する。バス上にトランザクションが発行された場合、そのバスに接続されたエージェント（CPUやバスブリッジ等）は、そのトランザクションの要求アドレスについて、最新データ（すなわち、ステートM）を持っているか、メモリと同内容のデータ（ステートE又はS）を持っているか、を示す信号をバス上に出力する。この信号をスヌープ出力という。以下、簡単のため、最新データを持っていることを示す信号をHITM、メモリと同内容のデータを持っていることを示す信号をHITという。より詳しく説明すれば、本実施形態では、バスにはHITM用の信号線とHIT用の信号線が設けられており、トランザクションを受け取ったエージェントは、要求されたブロックをステートMで持っている場合にはHITMをアサートし、要求されたブロックをステートE又はSで持っている場合にはHITをアサートする。そして、エージェントがトランザクションの要求アドレスに対応するブロックを持っていない場合（ステートIのとき）には、HITMもHITもアサートされず、この状態を「ミス」と呼ぶ。

【0113】トランザクション発行元のエージェントは、他のエージェントからのスヌープ出力のワイヤードORをスヌープ結果として受け取る。したがって、スヌープ結果がHITMの場合は、バス上に当該トランザクションの要求アドレスに対応するブロックをステートMで持っているエージェントが存在することを意味し、スヌープ結果がHITの場合は、バス上に当該トランザクションの要求アドレスに対応するブロックをステートE又はSで持っているエージェントが存在することを意味する。そして、スヌープ結果がミスの場合、すなわちHITMでもHITでもない場合は、バス上に当該トランザクションの要求アドレスに対応するブロックを持っているエージェントがないことを意味する。

【0114】表6に示すように、バスブリッジが下位バスにReadを発行した場合には、下位バスのスヌープ

る。この場合、トランザクション終了後のブリッジキャッシュのステート（以下、次ステートと呼ぶ）は、バスブリッジから下位バスに発行したトランザクションとこのトランザクションに対する下位バスのスヌープ結果に基づいて決定する。

【0111】表6は、ブリッジキャッシュの次ステートの決定規則を示す表である。

【0112】

【表6】

結果によってブリッジキャッシュの次ステートを変える。すなわち、下位バススヌープ結果がHIT又はHITMであったときは次ステートをSとし、下位バススヌープ結果がミスであったときは次ステートをEとする。一方、バスブリッジが下位バスにInvalidate系トランザクションを発行した場合には、下位バスのスヌープ結果によらず次ステートをMとする。

【0115】以上説明したようにブリッジキャッシュの次ステートを決めることにより、ブリッジキャッシュ同士の間のキャッシュ・コンシステンシを維持することができる。

【0116】[1. 3] スプリット応答時における上位バスへのスヌープ出力

バスブリッジが上位バスに発行されたトランザクションをスプリットした場合、そのバスブリッジは、スプリット時に下位バスに発行したトランザクションに対する他の下位バスエージェントからの応答を受け取り、その結果に基づきトランザクション発行元のCPUに対して応答を行う必要がある。バスブリッジからトランザクション発行元CPUへのこのような応答のことをスプリット応答トランザクションという。バスブリッジは、下位バスエージェントからの応答が揃ったところで上位バスにアービトレーションを行って上位バスの使用权を獲得し、上位バスに対し必要な情報を含んだスプリット応答トランザクションを発行する。本実施形態では、このスプリット応答トランザクションのときにバスブリッジから上位バスに対して所定の規則にしたがったスヌープ出力を発し、このスヌープ出力により発行元CPUのキャッシュのステートを制御する。

【0117】従来の一階層のキャッシュシステムについてのMESIプロトコルでは、CPUからバス上にトランザクションが発行されたときに、そのバスのスヌープ結果から当該CPUのキャッシュの次ステートを決定する規則が定められている。その規則によれば、CPUからバスにReadが発行された場合、スヌープ結果がミ

スの場合は当該CPUキャッシュのステートをEに変え、スヌープ結果がHIT又はHITMの場合は当該CPUキャッシュのステートをSに変える。また、その規則では、CPUからバスにInvalidate系トランザクションが発行された場合は、スヌープ結果にかかわらず当該CPUキャッシュのステートをMに変える。

【0118】本実施形態では、CPUキャッシュは従来の一階層キャッシュ用のMESIプロトコルにしたがっており、したがってCPUキャッシュのステータはその規則にしたがって遷移する。そして、本実施形態では、

その規則を利用してバスブリッジからのスヌープ出力によりその上位のCPUキャッシュのステートを制御することにより、制約1又は制約2の関係を維持する。

【0119】[1. 3. 1] 制約1を採用した場合
本実施形態において制約1を採用した場合には、バスブリッジは、スプリットを行った場合に、下位バスに発行したトランザクションの種類に基づき、表7の規則に従って上位バスに発行するスヌープ出力を決定する。

【0120】

〔表7〕

	下位バスのスヌープ結果		
下位バスに発行したトランザクション	ミス	HIT	HITM
Read	HIT (S)	HIT (S)	HIT (S)
Invalidate	ミス (M)	ミス (M)	ミス (M)
Read&Invalidate	ミス (M)	ミス (M)	ミス (M)

(注) 括弧 () はトランザクション発行元のCPUキャッシュの最終ステート

表7において、下位バスに発行したトランザクションがReadの場合、バスブリッジは下位バスのスヌープ結果にかかわらず上位バスにスヌープ出力としてHITをアサートする。下位バス・スヌープ結果がHIT又はHITMの場合は、他のブリッジキャッシュに当該ブロックがあるということなので、トランザクション発行元のCPUの次ステートをSにする必要がある。ここでCPUキャッシュは前述した一階層キャッシュに対するMESIプロトコルにしたがってステータ遷移するので、次ステートをSにするにはスヌープ結果としてHITを受け取る必要がある。そこで、バスブリッジはスプリット応答トランザクション時に上位バスに対してHITをアサートし、上位バス上のトランザクション発行元CPUは、そのHITを受けてCPUキャッシュの対応キャッシュブロックのステートをSとする。また、表7の規則では、下位バス・スヌープ結果がミスのときにも上位バスにHITをアサートすることにより、表1に示した制約1を満足するようにしている。すなわち、下位バス・スヌープ結果がミスのときには、他のブリッジキャッシュに当該ブロックを持っているものはないので、ブリッジキャッシュの当該ブロックの次ステータはEとなる(表6参照)。したがって、この場合に制約1を満たすためには、トランザクション発行元CPUのキャッシュの次ステータをSにする必要がある。このため、表7の

規則では、下位バス・スヌープ結果がミスの場合にも上位バスに対してHITをアサートする。

【0121】また、下位バスに発行したトランザクションがInvalidate系の場合には、バスブリッジは上位バスに対してHITもHITMもアサートしない(すなわち、ミス)。この場合、スプリット前の上位バスのトランザクションもInvalidate系であるので、トランザクション発行元のCPUキャッシュの次ステータはMとなる。このとき、ブリッジキャッシュの次ステータもMとなるので(表6参照)、この場合も制約1を満足する。

【0122】このように、表7の規則にしたがってスプリット応答時のスヌープ出力を生成することにより、スプリットを行った場合においても前記制約1を満足することができ、この結果MLIを満たし、キャッシュ・コンシステンスを維持することができる。

【0123】[1. 3. 2] 制約2を採用した場合
本実施形態において制約2を採用した場合には、バスブリッジは、スプリットを行った場合に、下位バスに発行したトランザクションの種類に基づき、表8の規則に従って上位バスに発行するスヌープ出力を決定する。

【0124】

〔表8〕

	下位バスのスヌープ結果		
下位バスに発行したトランザクション	ミス	HIT	HITM
Read	ミス (E)	HIT (S)	HIT (S)
Invalidate	ミス (M)	ミス (M)	ミス (M)
Read&Invalidate	ミス (M)	ミス (M)	ミス (M)

(注) 括弧 () はトランザクション発行元のCPUキャッシュの最終ステート

表8の規則と前記表7に示した制約1の場合の規則との相違は、バスブリッジから下位バスに発行したトランザ

クションがReadで下位バスのスヌープ結果がミスの場合のスヌープ出力の決め方である。すなわち、表8の

規則では、このような場合のバスブリッジから上位バスへのスヌープ出力をミスとしている。

【0125】バスブリッジから下位バスに発行したトランザクションがReadで下位バスのスヌープ結果がミスの場合、ブリッジキャッシュの次ステートはEとなるが、制約2ではブリッジキャッシュのステートがEの場合、その上位のCPUキャッシュのステートがEとなることを許す。したがって、この場合におけるバスブリッジからのスプリット応答トランザクションにおいてスヌープ出力をミスとすることにより、このスヌープ出力を受け取ったトランザクション発行元のCPUのCPUキャッシュの次ステートをEとすることができる。

【0126】このように、表8にしたがってスプリット応答時のスヌープ出力を生成することにより、スプリットを行った場合においても前記制約2を満足することができ、この結果MLIを満たし、キャッシュ・コンシステンシを維持することができる。

【0127】[2] 下位バス上に発行されたトランザクションに対する基本動作

ブリッジキャッシュステート	下位バス・トランザクション		
	Read	Read&Invalidate Invalidate	WriteBack
M	Read	Invalidate	X
E	○	Invalidate	X
S	○	Invalidate	X
I	○	○	○

○：上位バスにトランザクションを発行しない
×：あり得ない

すなわち、下位バス・トランザクションがReadの場合、そのトランザクションを受け取ったバスブリッジは、自分のブリッジキャッシュのステートがMの場合にのみ上位バスに対してReadを発行する。ブリッジキャッシュのステートがMの場合、そのバスブリッジの上位のCPUが当該ブロックの最新データを持っている可能性があるからである。このようにしてバスブリッジから上位バスにReadが発行された場合、そのReadを受け取ったCPUがそのReadの要求するキャッシュブロックをMステートで持っていれば、そのCPUは上位バスに対してスヌープ出力としてHITMをアサートすると共に、当該Mステートのブロックをバスブリッジに返す。このとき、そのCPUのキャッシュのステートはSに変化する。これを受け取ったバスブリッジは、このブロックを下位バスを介して下位バス・トランザクション発行元のバスブリッジに返す。そして、この下位バス・トランザクション発行元のバスブリッジは、下位バスから受け取った最新のデータをスプリット応答としてReadの発行元のCPUに返す。

【0131】下位バス・トランザクションがReadの場合においてブリッジキャッシュのステートがM以外の場合は、ここではCPUキャッシュとブリッジキャッシュは制約1を満足しているため、当該ブリッジキャッシ

他のバスブリッジから下位バス上に発行されたトランザクションを受け取ったバスブリッジの一連の動作について説明する。すなわち、ここでは、他のバスブリッジが自分の上位のCPUからのトランザクションをスプリットして下位バスにトランザクションを発行した場合に、その下位バス・トランザクションを受け取ったバスブリッジの動作を説明する。

【0128】[2.1] 上位バス出力判定

下位バス・トランザクションを受け取ったバスブリッジは、その下位バス・トランザクションの種類と現在のブリッジキャッシュのステートとに基づき、上位バスに発行すべきトランザクションを決定する。

【0129】[2.1.1] 制約1を採用する場合

制約1を採用する場合には、下位バス・トランザクションを受け取ったバスブリッジは、次の表9に示す規則にしたがって、上位バスに発行するトランザクションを決定する。

【0130】

【表9】

ユの上位のCPUキャッシュには要求ブロックをステートMで持っているものはない。したがって、バスブリッジは上位バスに対してトランザクションを発行する必要がない。

【0132】また、下位バス・トランザクションがInvalidateまたはRead&Invalidateの場合、そのトランザクションを受け取ったバスブリッジは、自分のブリッジキャッシュのステートがM、E、Sの場合に上位バスに対してInvalidateを発行する。すなわち、ブリッジキャッシュのステートがM、E、Sの場合、その上位のCPUキャッシュが無効化すべきデータを持っている可能性があるため、バスブリッジは上位バスにInvalidateを発行する。一方、ブリッジキャッシュのステートがIの場合は、その上位のCPUキャッシュにはトランザクションの対象となるブロックがないため、上位バスにトランザクションを発行する必要がない。

【0133】そして、下位バス・トランザクションがWriteBackの場合は、バスブリッジは、上位バスにトランザクションを発行しない。バスブリッジが下位バスにWriteBackを発行するのは、自分のブリッジキャッシュのブロックをエビクションしてメインメモリに書き戻す場合であり、WriteBackの発行

元は他のバスブリッジに対してデータの要求も無効化の要求もしていないからである。

【0134】以上説明した規則にしたがってバスブリッジが上位バスに発行するトランザクションを決定することにより、ブリッジキャッシュとCPUキャッシュとの制約1（表1）を満足させ、キャッシュ・コンシステンスを維持することができる。

【0135】[2. 1. 2] 制約2を採用する場合

制約2を採用する場合、バスブリッジは、下位バスのトランザクションを受け取ると、そのトランザクションの種類とそのトランザクションを受け取ったときのブリッジキャッシュのステートとに基づき、表10に示す規則にしたがって上位バスに発行するトランザクションを決定する。

【0136】

【表10】

ブリッジキャッシュステート	下位バス・トランザクション		
	Read	Read&Invalidate invalidate	WriteBack
M	Read	Invalidate	X
E	Read	Invalidate	X
S	○	Invalidate	X
I	○	○	○

○：上位バスにトランザクションを発行しない
X：あり得ない

表10の規則は、下位バスから受け取ったトランザクションがReadでかつブリッジキャッシュのステートがEである場合において上位バスへReadを発行する点が、制約1における表9の規則と異なる。

【0137】制約2では、ブリッジキャッシュのステートがEのときに、その上位のCPUキャッシュがステートMとなっている（すなわち、CPUキャッシュが最新のデータを持っている）可能性がある。したがって、下位バスから受け取ったトランザクションがReadでかつブリッジキャッシュのステートがEである場合には、バスブリッジは上位バスに対してReadを発行する。もしそのバスブリッジの上位のCPUキャッシュの一つがReadが要求するデータをステートMで持っていれば、そのCPUは上位バスに対してスヌープ出力としてHITMをアサートすると共に、当該Mステートのブロックをバスブリッジに返す。この結果、バスブリッジは最新のデータを有することとなる。そして、バスブリッジは、最初に受け取った下位バス・トランザクションに対する応答として、この最新データを下位バスに出力する。

【0138】なお、表10におけるこれ以外の場合についての規則は、表9の規則と同様である。

【0139】以上説明した表10の規則にしたがってバスブリッジが上位バスに発行するトランザクションを決定することにより、表2に示したブリッジキャッシュとCPUキャッシュとの制約2を満足させ、キャッシュ・コンシステンスを維持することができる。

【0140】なお、上記表9又は表10の規則にしたがってバスブリッジから上位バスにトランザクションが発行された場合、その上位バスに接続された各CPUは、階層を意識せず、そのトランザクションが他のCPUから発行された場合と同様に応答する。したがって、例えばバスブリッジから上位バスにInvalidateが発行されたときに、その上位バス上にInvalida

teの対象ブロックをステートMで持っているCPUがあった場合は、そのCPUは上位バスに対してHITMをアサートすると同時にその対象ブロックのデータを上位バスに出力する。

【0141】—[2. 1. 3]—変形例—

[2. 1. 1] 及び [2. 1. 2] で説明した上位バス出力判定方式によれば、下位バス・トランザクションがInvalidateの場合、これを受け取ったバスブリッジのブリッジキャッシュのステートがMであれば、当該バスブリッジは、原則として上位バスに対してInvalidateを発行する（表9及び表10参照）。

【0142】ところが、あるCPUキャッシュにおいてエビクションが発生し、ステートMのブロックがそのCPUの下位のブリッジキャッシュに書き戻された場合、そのブリッジキャッシュのステートはMとなるのに対し、元のCPUキャッシュと同じ上位バスに接続されたCPUキャッシュはすべてステートIとなる。したがって、この場合、バスブリッジは、CPUから書き戻されたブロックに対するInvalidateを下位バスから受け取ったとしても、上位バスに対してInvalidateを発行する必要はない。表9又は表10に示した方式においては、このような場合でも上位バスに対するInvalidateを発行することとなり、不要なInvalidateの発行を行っていることとなる。

【0143】また、[2. 1. 1] 及び [2. 1. 2] で説明した上位バス出力判定方式によれば、下位バス・トランザクションがReadの場合、これを受け取ったバスブリッジのブリッジキャッシュのステートがMであれば、当該バスブリッジは、原則として上位バスに対してReadを発行する（表9及び表10参照）。

【0144】ところが、あるCPUがステートMのブロックを持っているときに、そのCPUと同一上位バス上のCPUがそのブロックをReadした場合、当該上位バス上のCPUキャッシュのステートはS又はIとな

り、それらCPUキャッシュの下位のブリッジキャッシュのステートはMとなる。すなわち、当該ブロックの最新のデータはブリッジキャッシュが所有していることとなる。この場合、バスブリッジは、当該ブロックに対するReadを下位バスから受け取ったとしても、上位バスに対してReadを発行する必要はない。表9又は表10に示した方式においては、このような場合でも上位バスに対するReadを発行することとなり、不要なReadの発行を行っていることとなる。

【0145】この変形例は、以上に説明したような不要なトランザクションの発行を抑制することを目的とする。

【0146】このため、本変形例では、バスブリッジにおいて、ブリッジキャッシュの各ブロックごとに状態フラグを設け、この状態フラグによってトランザクションの発行を制御する。

【0147】図2は、この変形例の概略構成を示す図である。図2において、上位バス20と下位バス40との間でトランザクションの制御を行うバスブリッジ30は、バスブリッジ制御部60と上位キャッシュ状態フラグ記憶部80とを含む。上位キャッシュ状態フラグ記憶部80は、ブリッジキャッシュ31の各ブロックごとに状態フラグを有する。この状態フラグは2ビットであり、各ビット(A)、(B)は、それぞれ次の状態を示すためのビットである。

【0148】(A)そのフラグに対応するブリッジキャッシュのブロックをInvalidateするために、上位バス20上の各CPU10及び11のキャッシュをInvalidateする必要がある。すなわち、MESIプロトコルでは、上位バス20上の各CPUのキャッシュのステートがすべてIである。

(B)そのフラグに対応するブリッジキャッシュ31のブロックのデータを下位バス40に出力するために、上位バス20に対してReadを発行する必要がない。すなわち、上位バス20上の各CPUのキャッシュはI又はSである。

バスブリッジ制御部60は、上位バス20・下位バス40間のトランザクションのやり取り及びブリッジキャッシュ31の制御のほかに、上位キャッシュ状態フラグ記憶部80の制御を行う。

【0149】バスブリッジ制御部60は、CPU10又は11から上位バス20上に発行されたトランザクションをモニタする。そして、そのトランザクションの種類に基づき、上位キャッシュ状態フラグ記憶部80の当該ブロックに対する状態フラグを制御する。すなわち、バスブリッジ制御部60は、上位バス20上にWrite Backが発行されたのを検出したときには、上位キャッシュ状態フラグ制御部80における該当ブロックに対する状態フラグのビット(A)をセットする。また、バスブリッジ制御部60は、上位バス20上にReadが

発行されたのを検出したときには、上位キャッシュ状態フラグ制御部80における該当ブロックに対する状態フラグのビット(B)をセットする。そして、上位バス20上にInvalidate系トランザクションが発行された場合には、バスブリッジ制御部60は、該当ブロックに対する状態フラグを両ビットともリセットする(CPUがInvalidate系を発行するのは、該当ブロックに対して書き込みを行うときだからである)。

【0150】そして、バスブリッジ制御部60は、下位バス40からInvalidateを受け取った場合、まずブリッジキャッシュ31をスヌープしてInvalidate対象ブロックのステートを調べる。このステートがMの場合は、上位キャッシュ状態フラグ記憶部80を検索して、当該ブロックの状態フラグのビット

(A)を調べる。そして、このビット(A)がリセット状態である場合にのみ、バスブリッジ制御部60は上位バス20に対してInvalidateを発行する。ビット(A)がセット状態の場合は、上位バス20上のCPUキャッシュはすべてステートIなのでInvalidateの発行は行わない。

【0151】また、バスブリッジ制御部60は、下位バス40からReadを受け取った場合、ブリッジキャッシュ31をスヌープしてRead対象のブロックのステートを調べる。このステートがMの場合は、さらに上位キャッシュ状態フラグ記憶部80を検索して、当該ブロックの状態フラグのビット(B)を調べる。そして、このビット(B)がリセット状態の場合のみ、バスブリッジ制御部60は上位バス20に対してReadを発行する。

【0152】このように、図2の構成では、バスブリッジ制御部60において、上位バス20上のトランザクションをモニタすることにより、上位バス20上のCPUキャッシュの状態を把握する。そして、この結果得た情報を用いて下位バス40から上位バス20へのトランザクションの転送を制御することにより、上位バス20上へ発行されるトランザクションを減らすことができる。したがって、図2の構成によれば、上位バス20の負荷を低減することができる。

【0153】[2.2] 下位バスに対するスヌープ出力
下位バスからトランザクションを受け取ったバスブリッジが、表9又は表10の規則にしたがって上位バスにトランザクションを発行すると、上位バスの各CPUからそのトランザクションに対する応答が当該バスブリッジに返される。それら上位バスからの応答に基づき、バスブリッジは下位バスに対して応答を発する。この応答の時に、バスブリッジは、下位バス・トランザクション発行元のバスブリッジに対するスヌープ出力を下位バス上に発する。

【0154】ここでは、他のバスブリッジから下位バス

に発行されたトランザクションを受け取ったバスブリッジが、このトランザクションに対して下位バスに返すスヌープ出力の決定の方法について説明する。

【0155】 [2. 2. 1] 制約1を採用する場合
制約1を採用する場合、バスブリッジから下位バスへ発行するスヌープ出力は、当該バスブリッジが受け取った

下位バス・トランザクションの種類と、下位バストランザクションを受け取った時点でのブリッジキャッシュのステートとに基づき、次の表11の規則にしたがって決定する。

【0156】

【表11】

ブリッジキャッシュ ステート	下位バス・トランザクション		
	Read	Read&Invalidate Invalidate	WriteBack
M	HIT	HIT	×
E	HIT	ミス	×
S	HIT	ミス	×
I	ミス	ミス	ミス

×：あり得ない

まず、下位バス・トランザクションがReadであった場合について説明する。この場合においてブリッジキャッシュのステートがIの場合には、当該ブリッジキャッシュ及びその上位のCPUキャッシュのいずれもReadが要求するデータを持っていないので、バスブリッジは下位バスに対するスヌープ出力をミスとする。また、ブリッジキャッシュがS又はEの場合には、当該ブリッジキャッシュが下位バスのReadの要求するデータを持っており、しかもそのデータの内容はメインメモリと同内容なので、バスブリッジは下位バスに対するスヌープ出力をHITとする。特に、制約1を採用する場合は、ブリッジキャッシュがEの場合にはその上位のCPUキャッシュはMとなり得ないので、ブリッジキャッシュがEの場合はバスブリッジは上位バスをスヌープするまでもなく、下位バスに対してHITをアサートすれば足りる。また、ブリッジキャッシュのステートがMの場合は、ブリッジキャッシュ又はその上位のCPUキャッシュのいずれかがそのReadの要求するデータを持っているので、下位バスに対してHITMをアサートする。なお、ブリッジキャッシュのステートがMの場合、前述したようにバスブリッジから上位バスにReadが発行される。このReadに対する上位バスのスヌープ結果がHITMだった場合は、バスブリッジは上位バスのCPUキャッシュから最新のデータを得てブリッジキャッシュに格納すると共に下位バス上にその最新データを出力する。一方、Readに対する上位バススヌープ結果がHIT又はミスであった場合には、バスブリッジはブリッジキャッシュが有するデータを最新データとして下位バスに出力する。

【0157】 次に、下位バス・トランザクションがRead&InvalidateまたはInvalidateであった場合について説明する。この場合において、ブリッジキャッシュのステートがI、S又はEであった場合には、バスブリッジは下位バスに対するスヌープ出力をミスとし、ブリッジキャッシュのステートがMであった場合には下位バスに対するスヌープ出力をHITMとする。Invalidate系トランザクションは、

対象とするキャッシュブロックを無効化（すなわちIステートに変える）するトランザクションであるから、ブリッジキャッシュの当該ブロックの次ステートはIとなるので、下位バスに対するスヌープ出力もミスとすればよい。ただし、ブリッジキャッシュのステートがMであった場合には、そのブリッジキャッシュ（又は上位のCPUキャッシュ）の持つ最新のデータをRead発行元のバスブリッジに返しかつメインメモリにも書き戻す必要があるため、下位バスにはHITMをアサートする。

【0158】 あるバスブリッジから下位バスに発行されたトランザクションがWriteBackであった場合には、他のバスブリッジのキャッシュのステートはI以外にはないので、他のバスブリッジはスヌープ出力をミスとする。

【0159】 なお、表11に示した規則において特徴的なのは、バスブリッジから下位バスへのスヌープ出力を決定する際に、上位バスのCPUの情報が不要なことである。したがって、バスブリッジは、下位バスからトランザクションを受け取ると、そのトランザクションの種類とそのときのブリッジキャッシュとステートを調べるだけで、下位バスに対するスヌープ出力を生成できる。これは、制約1のごとくCPUキャッシュのステートを制限することのメリットの一つである。

【0160】 以上説明した規則に従ってバスブリッジから下位バスに対するスヌープ出力を生成することにより、キャッシュ・コンシステンスを維持することが可能となる。

【0161】 [2. 2. 2] 制約2を採用する場合
制約2を採用した場合、バスブリッジから下位バスへ発行するスヌープ出力は、当該バスブリッジが受け取った下位バス・トランザクションの種類と、そのトランザクションを受け取った時点でのブリッジキャッシュのステートと、上位バスのスヌープ結果とに基づき、次の表12の規則にしたがって決定する。

【0162】

【表12】

ブリッジキャッシュ ステート	下位バス・トランザクション				
	Read		Read&Invalidate Invalidate		WriteBack
	上位バス スヌープ 結果 HITM	上位バス スヌープ 結果 HITM 以外	上位バス スヌープ 結果 HITM	上位バス スヌープ 結果 HITM 以外	
M E S I	HITM HITM × ×	HITM KIT HIT ミス	HITM HITM × ×	HITM ミス ミス ミス	× × × ミス

×：あり得ない

表12の規則は、表11に示した制約1の場合の規則と異なり、上位バスのスヌープ結果によって下位バスへのスヌープ出力を区別している。ここでいう上位バスのスヌープ結果とは、下位バス・トランザクションについてバスブリッジが表10の規則にしたがって生成した上位バス・トランザクションに対する上位バスのスヌープ結果のことである。

【0163】表12の規則において、下位バス・トランザクションの種類がWriteBackの場合、及び下位バス・トランザクションの種類がRead、Invalidate又はRead&Invalidateの場合において上位バスのスプリット結果がHITM以外の場合のスヌープ出力の決定の規則は、表11の場合と同様である。

【0164】表12の規則の特徴点は、下位バス・トランザクションの種類がRead、Invalidate又はRead&InvalidateでブリッジキャッシュのステートがEの場合において、上位バスのスヌープ結果がHITMとなると、下位バスに対するスヌープ出力をHITMとする点である。

【0165】これは、制約2において、ブリッジキャッシュのステートがEの場合にはその上位のCPUキャッシュがステートMとなり得るためである。すなわち、ブリッジキャッシュのステートがEのときに、バスブリッジから上位バスに発行したトランザクションに対するスヌープ結果がHITMであった場合には、上位バス上のCPUがステートMのデータを持っていたことになり、この結果を下位バス・トランザクションの発行元のバスブリッジへ知らせるために、下位バスへのスヌープ出力をHITMとする。

【0166】以上説明した規則に従ってバスブリッジから下位バスに対するスヌープ出力を生成することにより、キャッシュ・コンシステンスを維持することが可能

となる。

【0167】[2. 3]ブリッジキャッシュの次ステート

次に、下位バスに発行されたトランザクションを受け取ったバスブリッジのブリッジキャッシュのステートの遷移について説明する。なお、ブリッジキャッシュの次ステートの決定の仕方は、制約1及び制約2のいずれを採用した場合も同じである。

【0168】まず、下位バス・トランザクションを受け取ったバスブリッジが上位バスに対してトランザクションを発行しない場合について説明する。

【0169】下位バス・トランザクションがReadであった場合において、それを受け取ったバスブリッジのブリッジキャッシュのステートがIであった場合には、そのブリッジキャッシュの次ステートはIとなる。また、Readを受け取ったブリッジキャッシュのステートがS又はEであった場合には、そのブリッジキャッシュの次ステートはSとなる。

【0170】下位バス・トランザクションがInvalidate系であった場合には、それを受け取ったバスブリッジのブリッジキャッシュの次ステートはIとなる。

【0171】下位バス・トランザクションがWriteBackであった場合には、それを受け取ったバスブリッジのブリッジキャッシュのステートは変化しない。

【0172】次に、下位バス・トランザクションを受け取ったバスブリッジが上位バスに対してトランザクションを発行する場合について説明する。この場合、ブリッジキャッシュの次ステートは、表13に示す規則にしたがって決定する。

【0173】

【表13】

	上位バスのスヌープ結果		
上位バスに発行した トランザクション	ミス	H I T	H I T M
Read	S	S	S
Invalidate	I	X	I
Read&Invalidate	I	X	I

X : あり得ない

表13は、バスブリッジが上位バスに発行したトランザクションとそのトランザクションに対する上位バスのスヌープ結果との各組み合わせに対するブリッジキャッシュの次ステートが示されている。

【0174】この規則では、バスブリッジが上位バスにReadを発行した場合には、上位バスのスヌープ結果にかかわらず、ブリッジキャッシュの次ステートをSとする。一方、バスブリッジが上位バスにInvalidate系トランザクションを発行した場合には、ブリッジキャッシュの次ステートをIとする。

【0175】このような規則に従えば、上位バスのスヌープ結果を用いずに容易にブリッジキャッシュの次ステートを決定することができる。

【0176】下位バス・トランザクションを受け取ったバスブリッジが上位バスに対してトランザクションを発行した場合における、ブリッジキャッシュの次ステートの決め方にはもう一つの方法がある。その決め方の規則を表14に示す。

【0177】

【表14】

	上位バスのスヌープ結果		
上位バスに発行した トランザクション	ミス	H I T	H I T M
Read	I	S	S
Invalidate	I	X	I
Read&Invalidate	I	X	I

X : あり得ない

この規則が表13の規則と異なる点は、上位バスにReadを発行した場合において、上位バスのスヌープ結果に基づいてブリッジキャッシュの次ステートを決定している点である。表14の規則では、上位バスにReadを発行した場合、上位バスのスヌープ結果がHIT又はHITMの場合には表13の規則と同様ブリッジキャッシュの次ステートをSとするが、上位バスのスヌープ結果がミスの場合はブリッジキャッシュの次ステートをIとする。このようにすることによって次のような効果が得られる。

【0178】すなわち、上位バスのスヌープ結果がミスということは、当該バスブリッジの上位バス上のCPUキャッシュが現在すべてステートIであるということの意味する。従って、このような場合にブリッジキャッシュの次ステートをIとしても、キャッシュ・コンシステンシは満足される。また、上位バス上のCPUキャッシュがすべてIということは、上位バス上のCPUはReadトランザクションが要求するブロックを現在全く使用していないということの意味するので、このような場合にブリッジキャッシュのみが当該ブロックを持っていたとしても無駄になる可能性が高い。そこで、表14に示したように上位バスのスヌープ結果がミスのときにブリッジキャッシュの次ステートをIとすることにより、ブリッジキャッシュに空きブロックが生まれ、キャッシュの有効利用を図ることができる。

【0179】以上、上位バスに発行されたトランザクシ

ョンを受け取ったときのバスブリッジの基本動作と、下位バスに発行されたトランザクションを受け取ったときのバスブリッジの基本動作とを説明した。次は、ブリッジキャッシュのエビクション時のバスブリッジの動作、及び上位バスのトランザクションに対するリトライについて説明する。

【0180】[3]ブリッジキャッシュのエビクションを行う場合のバスブリッジの動作

キャッシュメモリには様々な方式があるが、制御の容易さやコストなどからダイレクトマッピング方式やセットアソシアティブ方式が広く用いられている。これらの方式においては、一つのキャッシュブロックに対して複数のメモリアドレスを対応させているため、あるデータを読み込もうとする場合に、そのデータに対応するキャッシュメモリのブロックがそのブロックに対応する他のアドレスのデータによって使用され、空きがない場合がある。このような場合に、現在使用されているブロックを無効化して空きブロックを作る処理のことをエビクションと呼ぶ。キャッシュに読み込もうとしていたデータは、エビクションの結果できた空きブロックに格納される。

【0181】さて、ブリッジキャッシュにてエビクションを行った場合、上位バス上には当該エビクションを行ったブロックを持っているCPUキャッシュが存在する可能性がある。従って、ブリッジキャッシュのエビクションを行った場合には、上位バスのCPUキャッシュの

当該ブロックを無効化する必要がある場合がある。また、エビクション対象のブロックが最新のデータであった場合、そのデータをメインメモリに書き戻す必要がある。

【0182】本実施形態において、上位バスのトランザクションを検出したバスブリッジは、ブリッジキャッシュを検索して当該トランザクションのアドレスに対応するブロックが空いているか否かを調べ、空きブロックがない場合にエビクションを行う。エビクションにおいては、ブリッジキャッシュの上位のCPUキャッシュの該当ブロックを無効化し、ブリッジキャッシュの該当ブロックのデータをメインメモリへ書き戻したのち、ブリッジキャッシュの該当ブロックをステートをIに変更して使用可能な状態とする。

【0183】【3.1】上位バス上のCPUキャッシュの無効化

上位のCPUキャッシュ内のエビクション対象ブロックを無効化するためには、バスブリッジは上位バスに対してInvalidateを発行する。このInvalidateの発行は、次の表15に基づいて行う。

【0184】

【表15】

ブリッジキャッシュ ステート	上位バスに発行する トランザクション
M	invalidate
E	invalidate
S	invalidate
I	○

○：発行しない

すなわち、本実施形態では、バスブリッジは、自らのブリッジキャッシュにおけるエビクション対象のブロックのステートに基づき、上位バスにInvalidateを発行するか否かを決定する。なお、この表15及び次に説明する表16及び17では、ブリッジキャッシュのステートの意味が、これまでの表3や4などの場合と異なる。表3等において、ブリッジキャッシュのステートといった場合、トランザクションに対するステート、すなわちトランザクションの要求データがブリッジキャッシュに有るか無いか等を意味した。これに対して、表15等では、ブリッジキャッシュのステートとは、トランザクションとは無関係に、ブロック自体に格納されているデータのステートを意味する。

【0185】表15の規則に従う場合、バスブリッジは、ブリッジキャッシュのステートがM、E、Sの場合に上位バスにInvalidateを発行する。ブリッジキャッシュのステートがM、E、Sの場合、上位のCPUキャッシュが当該ブロックを持っている可能性があるからである。

【0186】なお、このInvalidateを受け取

った上位バスの各CPUは、自らのCPUキャッシュを検索してInvalidate対象のブロックの状態を調べ、上位バスに対してスヌープ出力を返す。またこのとき、上位バス上のCPUのなかに当該ブロックをステートMで所有しているものがあれば、そのCPUは当該ブロックのデータを上位バスに出力する。バスブリッジは、CPUから上位バス上に出力されたデータを受け取り、そのデータにより当該エビクション対象のブロックを更新する。

【0187】なお、表15の規則は、制約1及び制約2に共通のものである。

【0188】【3.2】ブリッジキャッシュのデータのメインメモリへの書き戻し

また、ブリッジキャッシュ内の最新データをメインメモリに対して書き戻すためには、バスブリッジは下位バスにWriteBackを発行する。

【0189】【3.2.1】制約1を採用する場合
制約1を採用する場合は、バスブリッジから下位バスへのWriteBackの発行は、表16の規則に基づいて行う。

【0190】

【表16】

ブリッジキャッシュ ステート	下位バスに発行する トランザクション
M	WriteBack
E	○
S	○
I	○

○：発行しない

バスブリッジは、ブリッジキャッシュのステートに基づいて、下位バスにWriteBackを発行するか否かを決定する。表16に示すように、バスブリッジは、ブリッジキャッシュがステートMのときにのみ、下位バスに対してWriteBackを発行する。ブリッジキャッシュがステートM以外の場合は、そのブリッジキャッシュ自体のもっているデータはメインメモリと同内容であり、また制約1（表1）を満足していればその上位のCPUキャッシュが当該ブロックの最新データを持っていることはないから、そのような場合にはデータの下記戻しは不要だからである。

【0191】したがって、ブリッジキャッシュがMのときには、Invalidateによって上位バス上のCPUからブリッジキャッシュに書き戻された当該ブロックの最新データ、もしくはそのような上位バスからの書き戻しがなかった場合はブリッジキャッシュ自身が持っているデータが、WriteBackトランザクションによってメインメモリに書き戻される。一方、ブリッジキャッシュがEまたはSのときには、バスブリッジはメインメモリへのデータの書き戻しは行わず、単に当該ブロックのステートをIに変更し、上位バスに当該ブロッ

クに対する *Invalidate* を発行する。ブリッジキャッシュのステートが *I* のときは、エビクション自体が行われず、従って上位バスに対しても下位バスに対してもトランザクションは発行されない。

【0192】 [3. 2. 2] 制約2を採用する場合
制約2を採用する場合、ブリッジキャッシュのステートが *E* のときに、そのブリッジキャッシュの上位のCPUキャッシュがステート *M* となっている可能性がある。したがって、ブリッジキャッシュのステートが *E* の場合には、バスブリッジは、上位バス上のCPUキャッシュを調べ、もしステート *M* のCPUキャッシュがあれば、そのCPUキャッシュからステート *M* のデータを取得し、これをメインメモリに書き戻す必要がある。上位バス上のCPUキャッシュがステート *M* であるか否かは、前記 [3. 1] で説明した *Invalidate* に対する上位バスからのスヌープ結果から知ることができる。すなわち、上位バス上のCPUキャッシュがステート *M* の場

合、そのCPUキャッシュは、バスブリッジから *Invalidate* を受け取ると、スヌープ出力として *HITM* をアサートするので、バスブリッジの上位バス・スヌープ結果は *HITM* となる。

【0193】 なお、ステート *M* のCPUキャッシュは、バスブリッジから *Invalidate* を受け取ると、応答としてステート *M* のデータを上位バス上に出力するので、バスブリッジは上位CPUキャッシュからのステート *M* のデータを自動的に取得することができる。

【0194】 したがって、制約2を採用する場合におけるバスブリッジから下位バスへの *WriteBack* の発行は、ブリッジキャッシュのステート及び *Invalidate* に対する上位バスのスヌープ結果とに基づき、次の表17の規則に従って行う。

【0195】

【表17】

ブリッジキャッシュ ステート	下位バスに発行するトランザクション	
	上位バスのスヌープ 結果 <i>HITM</i>	上位バスのスヌープ 結果 <i>HITM</i> 以外
<i>M</i>	<i>WriteBack</i>	<i>WriteBack</i>
<i>E</i>	<i>WriteBack</i>	○
<i>S</i>	○	○
<i>I</i>	○	○

○：発行しない

制約2を採用した場合、ブリッジキャッシュのステートが *E* 又は *M* のときに、上位のCPUキャッシュがステート *M* になる可能性がある。したがって、ブリッジキャッシュのステートが *E* 又は *M* の場合に、上位バス・スヌープ結果が *HITM* になる可能性がある。したがって、表17では、ブリッジキャッシュのステートが *E* 又は *M* で、上位バス・スヌープ結果が *HITM* の場合には、下位バスに対して *WriteBack* を発行する。この *WriteBack* によってブリッジキャッシュからメインメモリに書き戻されるデータは、バスブリッジ（及びブリッジキャッシュ）が上位のCPUキャッシュから取得したステート *M* のデータである。

【0196】 また、ブリッジキャッシュのステートが *M* の場合であっても、上位のCPUキャッシュがすべてステート *M* 以外の場合もある。このような状態は、例えば、あるCPUキャッシュのステート *M* のデータがエビクションによりブリッジキャッシュに書き戻された場合などに起こる。この場合、*Invalidate* に対する上位バスのスヌープ結果は *HITM* 以外となる。このとき、最新のデータはブリッジキャッシュが所有していることになる。したがって、表17において、ブリッジキャッシュのステートが *M* で、上位バス・スヌープ結果が *HITM* 以外の場合には、バスブリッジは下位バスに

対して *WriteBack* を発行する。このとき、ブリッジキャッシュからメインメモリに書き戻されるデータは、ブリッジキャッシュが所有しているデータである。

【0197】 なお、ブリッジキャッシュがステート *E* で、上位バス・スヌープ結果が *HITM* 以外の場合には、上位のCPUキャッシュがステート *M* である可能性はないので、この場合には *WriteBack* は発行しない。

【0198】 以上説明した規則にしたがって上位バスへの *Invalidate* の発行及び下位バスへの *WriteBack* の発行を行うことにより、システム全体のキャッシュ・コンシステンスを維持しつつブリッジキャッシュのエビクションを行うことができる。

【0199】 [3. 3] トランザクション処理とエビクション処理との関係

次に、本実施形態におけるブリッジキャッシュのエビクションのためのバスブリッジの処理動作と、そのエビクションの起因となった上位バス・トランザクションのためのバスブリッジの処理動作との関係について説明する。

【0200】 前述したように、上位バス・トランザクションのアドレスに対応するブリッジキャッシュのブロックが空きブロックにならない限り、バスブリッジはその

上位バス・トランザクションの処理を進めることはできない。

【0201】(a)そこで、まず考えられるのは、エビクションを先に行って空きブロックを確保した後、上位バス・トランザクションについての処理を行うという方式である。

【0202】エビクション処理においては、上位CPUキャッシュの無効化のためバスブリッジからInvalidateを発行する必要があるので、この時点で上位バスを使用しているトランザクション発行元CPUに上位バスをいったん解放させる必要がある。このために、本実施形態では、ブリッジキャッシュのエビクションが必要な場合には、トランザクション発行元のCPUにトランザクションのリトライを指示して、いったん上位バスを解放させる。リトライとは、現在発行中のトランザクションをいったん終了してバスを解放した後、所定時間経過後に再び同じトランザクションを発行する処理である。なお、この場合、CPUはリトライ機能を有するものを用いる。

【0203】このときの処理の流れは以下のようになる。まず、上位バス・トランザクションを受け取ったバスブリッジは、ブリッジキャッシュをスヌープしてエビクションが必要か否かを判定する。この結果、エビクションの必要があると判定した場合は、上位バスに対してリトライ終了信号を出力してトランザクション発行元CPUにリトライを指示し、エビクション処理(上位CPUのInvalidate及びメインメモリへのWriteBack)を行う。一方、バスブリッジからのリトライ終了信号を受け取ったトランザクション発行元CPUは、いったんトランザクションを終了して上位バスを解放し、その後所定時間経過後に同じトランザクションを再発行する。この再発行時には、バスブリッジにおけるエビクション処理が終了しており、バスブリッジは当該トランザクションの処理を進めることができる。

【0204】(b)以上説明したようなリトライを行う処理では、CPUが最初にトランザクションを発行してから、最終的に当該トランザクションが完結するまでの時間が長くなってしまう。また、ブリッジキャッシュのエビクションが完了した後CPUからトランザクションが再発行されるまでの間に、エビクションにより設けた空きブロックが他のCPUからのトランザクションにより先に使用されてしまう可能性もあり、このような場合は再びエビクションを行わなければならない場合がある。

【0205】このような場合に対処するための方法としては、バスブリッジ内に特別のバッファを設け、エビクション対象のブロックをそのバッファに一時的に退避してブリッジキャッシュを空け、先にCPUからのトランザクションの処理を行ったのちにそのバッファに退避したブロックのデータを用いてエビクション処理を行うと

いう方式がある。

【0206】図3は、この方式におけるバスブリッジの構成を説明するための概略構成図である。図3において、上位バス20と下位バス40との間のトランザクションの制御を行うバスブリッジ30は、これまで説明したバスブリッジ基本動作を実行するバスブリッジ制御部60と、エビクション対象のキャッシュブロックを一時的に格納するエビクションバッファ62とを含む。エビクションバッファ62は、退避ブロックのデータ内容を格納するデータバッファ部66と、退避ブロックのアドレスを格納するアドレスレジスタ部64とを有する。なお、エビクションバッファ62は、数ブロック分のデータ及びアドレスを記憶することができ、複数のトランザクションの並行処理に対応可能となっている。エビクションバッファ62は必要な容量が小さいのでサイズも小さく、バスブリッジ制御部60と同一のチップの上に作り込むことができる(ちなみに、ブリッジキャッシュ31はMLIを満たすためにサイズが大きくなるため、バスブリッジ制御部60と同一のチップ上に形成することは困難である)。

【0207】図3において、上位バス20に発行されたトランザクションをバスブリッジ30が受け取った場合、バスブリッジ制御部60は、前述のようにブリッジキャッシュ31をスヌープしてトランザクションをスプリットすべきかなどの判定を行う。このとき、バスブリッジ制御部60は、トランザクションのアドレスに対応するブリッジキャッシュ31内のブロックの使用状況を検出し、検出の結果当該ブロックが空いていないことが分かると、当該ブロックのデータをブリッジキャッシュ31から読み出してエビクションバッファ62のデータバッファ部66に格納し、当該ブロックのアドレスをアドレスレジスタに登録する。これ同時に、バスブリッジ制御部60は、ブリッジキャッシュ31の当該ブロックをステートIに変更して使用可能な状態とする。

【0208】このような処理の結果、上位バス・トランザクションの処理が可能となるので、バスブリッジ制御部60は、前記[1]及び[2]節で説明した一連の処理を行って、そのトランザクションを完結させる。すなわち、バスブリッジ制御部60は、トランザクションのスプリットを行って下位バス40にトランザクションを発行し、この結果下位バス40の他のエージェント(メインメモリ又はバスブリッジ)から得たデータを、スプリット応答トランザクションとして上位バス20に出力する。

【0209】そして、この後、バスブリッジ制御部60は、エビクションバッファ62に一時退避したブロックに対するエビクション処理を行う。すなわち、バスブリッジ制御部60は、退避ブロックに対するInvalidateトランザクションを上位バス20に発行してCPUキャッシュとブリッジキャッシュとのコンシステン

シの維持を図り、当該退避ブロックがMステートである場合には下位バス40にWriteBackトランザクションを発行し、メインメモリに当該退避ブロックを書き戻す。なお、この場合において、バスブリッジ制御部60は、退避ブロックに対するInvalidateに対して、上位バス上のCPUからMステートのブロックが書き戻されてきた場合は、そのブロックのアドレスを判別して、この書き戻されてきたブロックをブリッジキャッシュ31ではなくエビクションバッファ62に書き込む。この場合、上位バスから書き戻されてきたデータは、最初にブリッジキャッシュ31から退避したデータの上に上書きされる。したがって、エビクションバッファ62のブロックは、メインメモリに書き戻すときには、必ず最新のデータとなっている。

【0210】このように、図3の構成によれば、トランザクションに対する処理をエビクションに先行して行うことができるため、CPUが最初にトランザクションを発行してからそのトランザクションが完結するまでの時間が大幅に短縮される。

【0211】なお、以上の説明では、図3の構成における実質的なエビクションの処理はトランザクションの処理が完結した後に行うとしていたが、この順序は必須のものではない。例えば、バスにおいてパイプライン処理が可能である場合には、上位バスが使用されていなければInvalidateを発行することができ、また上位バスのInvalidateが完了すればいつでも下位バスにWriteBackを発行することができる。したがって、この場合には、トランザクションに対する処理が完結する前にInvalidateやWriteBackを発行することも可能である。

【0212】また、図3の構成では、エビクション対象のブロックの退避場所としてエビクションバッファ62を用いたが、ブロック退避場所としては、図4に示すような補助キャッシュ68を用いることもできる。図4において、補助キャッシュ68は、数ブロックを格納することができる小容量のフルアソシアティブ・キャッシュである。図3のエビクションバッファ62と図4の補助キャッシュ68との違いは、エビクションバッファ62に格納されたブロックはバスブリッジ制御部60から読み出すことはできないが、補助キャッシュ68に格納されたブロックはバスブリッジ制御部60から読み出すことができるという点である。したがって、バスブリッジ制御部60が退避中のブロックを要求するトランザクションを上位バスから受け取った場合、図3の構成ではバスブリッジ制御部60はそのトランザクションに対する応答を行うことができないが、図4の構成によれば、補助キャッシュ68から当該ブロックを読み出して応答することができる。ただし、補助キャッシュ68の制御はエビクションバッファ62の制御よりも複雑となるため、図4のバスブリッジ制御部60の回路構成は、図3

のそれよりも複雑かつ大規模になる。

【0213】[4] リトライ処理

前記[3. 3]において、ブリッジキャッシュのエビクションが必要な場合におけるトランザクションのリトライについて説明したが、ここではその他にリトライを行う場合についての構成について説明する。

【0214】[4. 1] スプリット中に上位バスに別のトランザクションが発行された場合

バスブリッジが上位バス・トランザクションをスプリットし、スプリットにより下位バスに発行したトランザクションに対する下位バスからの応答がバスブリッジに来る前に、別のトランザクションが上位バスに発行された場合、先に発行されたトランザクションが使用する予定のキャッシュブロックが後に発行されたトランザクションによって使用されてしまい、前に発行されたトランザクションに対する下位バスからの応答をブリッジキャッシュに格納できなくなる可能性がある。

【0215】このような問題を解消するためにまず考えられるのは、バスブリッジにスプリット中か否かを示すフラグを設け、上位バス・トランザクションを受け取ったときにフラグがスプリット中を示していれば、そのトランザクションのスプリットを認めないようにすることである。すなわち、バスブリッジは、トランザクションを受け取ると、フラグを調べるとともにブリッジキャッシュをスヌープし、スプリット中かつ受け取ったトランザクションのスプリットが必要と分かった場合に、トランザクションをリトライさせる。このようにすれば、スプリット中のトランザクションのために確保されたキャッシュブロックが、後から発行されたトランザクションによって使用されるのを防ぐことができる。

【0216】しかしながら、この方法では、他のトランザクションのスプリット中に発行されたトランザクションは、必ずリトライするので無駄が多い。そこで、リトライを効率的に行う方法として、本実施形態では次の2つの方法を採用する。

【0217】第1の方法は、スプリット中にトランザクションが発行された場合、そのトランザクションの種類から、そのトランザクションをスプリットしたときにブリッジキャッシュのブロックを使用するかどうかを判定し、ブロックを使用する場合にのみリトライを行うという方法である。例えば、Readトランザクションは、スプリットした場合に、下位バスからの応答データを受け入れるためブロックを使用するが、Invalidateトランザクションは、スプリットしたとしてもブロックを使用しない。このようなトランザクションの種類によってリトライを行うか否かを判別することにより、無駄なリトライを減らすことができる。

【0218】この構成では、バスブリッジは、まず上位バス・トランザクションを受け取ると、フラグを調べて現在他のトランザクションがスプリット中か否かを判定

する。そして、スプリット中の場合は、上位バスから受け取ったトランザクションの種類を検出して、その種類がブロックを使用するものか否かを判定する。この結果、ブロックを使用するトランザクションである場合には、そのトランザクションのスプリットを認めない。すなわち、この場合、ブリッジキャッシュのスヌープによりそのトランザクションのスプリットが必要と判定されると、バスブリッジから当該トランザクションについてリトライ終了信号を返す。

【0219】この構成によれば、キャッシュブロックを使用することのないトランザクションは、リトライせずにそのまま実行することができる。

【0220】また、第2の方法は、スプリット中のトランザクションの使用予定ブロックのアドレスをバスブリッジで管理し、使用予定ブロックに対するトランザクションのみリトライするという方法である。すなわち、この方法では、スプリット中に新たに発行されたトランザクションが、現在スプリット中のトランザクションが使用する予定のブロックを使用するか否かを判定し、使用すると判定される場合には新たに受け取ったトランザクションに対してリトライを指示する。

【0221】図5は、このようなリトライ判定機能を有するバスブリッジの概略構成を示した図である。図5において、図3と同一の部分には同一の符号を付してその説明を省略する。

【0222】図5においては、バスブリッジ30内に現在スプリット中のトランザクションが使用するキャッシュブロックのアドレス情報を格納するスプリット情報レジスタ72が設けられている。

【0223】まず、図5の構成におけるスプリット情報レジスタ72へのアドレス情報の登録動作について説明する。この構成において、バスブリッジ制御部60は、上位バス20から受け取ったトランザクションをスプリットすると決定した場合、当該トランザクションのアドレスに対応するキャッシュブロックのアドレス情報を求め、スプリット情報レジスタ72に登録する。スプリット情報レジスタ72に登録されるアドレス情報は、例えばブリッジキャッシュ31がセットアソシアティブ方式である場合は、該当ブロックのインデックス及びウエイ番号である。そして、スプリットにより下位バス40に発行したトランザクションに対する応答が戻り、バスブリッジ制御部60から上位バス20に対してスプリット応答トランザクションを発行した時点で、当該トランザクションに対するスプリット情報レジスタ72の登録情報は消去される。

【0224】したがって、スプリット情報レジスタ72には、現在スプリット中のトランザクションが使用する予定のブロックのアドレス情報が格納されていることになる。なお、スプリット情報レジスタ72は複数のアドレス情報が登録可能であり、同時に複数のトランザクシ

ョンのスプリットに対応可能となっている。

【0225】次に、スプリット情報レジスタ72を用いた上位バス・トランザクションのリトライ判定について説明する。

【0226】上位バス・トランザクションを受け取ったバスブリッジ制御部60は、そのトランザクションのアドレスから、対応するキャッシュブロックのアドレス情報（例えば、インデックス及びウエイ番号）を求め、このアドレス情報をスプリット情報レジスタ72に登録されているアドレス情報と比較する。そして、この比較の結果、両者が一致した場合には、バスブリッジ制御部60は、上位バス・トランザクションに対する応答としてリトライ終了信号を返す。一方、比較の結果両者が一致しない場合には、バスブリッジ制御部60は、通常のトランザクション処理を行う。

【0227】このように、図5に示すバスブリッジ30によれば、スプリット中のトランザクションが使用する予定のキャッシュブロックに対するトランザクションは、リトライせずにそのまま実行することができる。

【0228】なお、スプリット情報レジスタ72にスプリットしたトランザクションのアドレス情報をすべて登録するかわりに、スプリットしたトランザクションのうちブリッジキャッシュを使用するトランザクションを選別し、このようなトランザクションのアドレス情報のみを登録する構成とすれば、無駄なリトライをさらに減らすことができる。なお、ブリッジキャッシュを使用するトランザクションは、トランザクションの種類から判別する。

【0229】また、前記第1及び第2の方法を併用すればリトライのさらなる効率化を達成することができる。

【0230】〔4.2〕エビクションバッファに退避中のデータに対するトランザクションが発行された場合この状況は、前記〔3.3〕(b) (図3参照)で説明したエビクションバッファを用いた構成において生じる。すなわち、エビクション対象のブロックをエビクションバッファに退避し、まだエビクションの処理が完了しないうちに、退避したデータに対するトランザクションが上位バスに発行された場合、エビクションバッファに格納されたデータにはアクセス不可能なので、そのような退避データに対するトランザクションはリトライする必要がある。この場合、エビクションバッファにデータを退避させている間に上位バスに発行されたトランザクションをすべてリトライさせるという方法もあるが、これではリトライが多くなり処理効率が向上しない。

【0231】そこで、本実施形態では、このような場合のリトライの発行を効率化するために、バスブリッジに、エビクションバッファ内のデータのアドレスとトランザクションのアドレスとを比較する手段を設ける。そして、比較の結果、トランザクションのアドレスと一致するアドレスのデータがエビクションバッファに格納さ

れているとわかった場合には、そのトランザクションをリトライさせる。

【0232】図3を参照して、この構成におけるバスブリッジの動作を説明する。上位バス20にトランザクションが発行された場合、バスブリッジ制御部60は、そのトランザクションのアドレスと、エビクションバッファ62のアドレスレジスタ64に登録されているアドレスとを比較する（このアドレスの比較は、ブロック単位の比較、すなわちタグ、インデックスまでの比較である）。この結果、トランザクションのアドレスと同一のアドレスがアドレスレジスタに登録されていた場合には、バスブリッジ制御部60は上位バス20に対してリトライ終了信号を出力する。

【0233】この構成によれば、エビクションバッファを採用した構成において、そのエビクションバッファに退避中のデータに対するトランザクションのリトライ処理を効率的に行うことができる。

【0234】[4. 3] 下位バスからのリトライ指示に対するバスブリッジの対応

リトライの指示は、下位バスのI/O装置などからも発せられる。このような下位バスからのリトライ指示をバスブリッジが受け取った場合、バスブリッジが所定時間後にトランザクションの再発行を行うという構成も考えられるが、この場合、バスブリッジの構成が複雑となってしまう。

【0235】そこで、バスブリッジの構成を簡単化するためには、バスブリッジは下位バスからリトライ終了信号を受け取った場合に、そのリトライ終了信号を上位バスに転送する構成を採用することもできる。すなわち、バスブリッジは、下位バスからリトライ終了信号を受け取った場合、スプリット応答トランザクションを生成して上位バスに発行し、そのスプリット応答トランザクションにてリトライ終了信号を出力する。そして、そのリトライ終了信号を受け取ったトランザクション発行元CPUが、当該トランザクションのリトライを行う。

【0236】この構成によれば、下位バスからのリトライ指示に対するバスブリッジの処理が簡単化される。

【0237】以上、本実施形態の階層バスシステムの動作について説明した。以下では、本実施形態におけるバスブリッジの構成及びその動作について説明する。

【0238】図6は、本実施形態におけるバスブリッジ及びブリッジキャッシュの構成を示すブロック図である。

【0239】図6において、バスブリッジ300は、信号線500～510によって上位バス20と信号のやり取りを行い、信号線512～522によって下位バス40と信号のやり取りを行う。

【0240】また、バスブリッジ300のブリッジキャッシュは、図6においては、タグRAM402及びキャッシュRAM404から構成されている。キャッシュR

AM404は、データをブロック単位で格納する。メモリから読み込まれたデータは、そのメモリアドレスのインデックス部に対応するアドレスのブロックに格納される。一方、タグRAM402は、キャッシュRAM404の各ブロックごとにメモリアドレスのタグ部を格納しており、キャッシュRAM404に各ブロックに格納されているデータがどのメモリアドレスのデータであるかを示している。また、タグRAM402は、各ブロックごとに、そのブロックに格納されているデータのMESIステートを保持している。

【0241】次に、図7に示す構成の階層バスシステムを例にとって、図6に示したバスブリッジ及びブリッジキャッシュの動作例を説明する。ここで、図7のバスブリッジ#0及び#1と、ブリッジキャッシュ#0及び#1とは、それぞれ図6に示す構成を有しているものとする。また、この例においては、ブリッジキャッシュはダイレクトマッピング方式のキャッシュであるとして説明する。

【0242】この動作例の初期状態を以下に示す。

【0243】

CPU#0: アドレスAのデータを持っていない（アドレスAに対応するCPUキャッシュのキャッシュブロックは無効化されている）

ブリッジキャッシュ#0: アドレスAのデータを持っていない。アドレスAに対応するブロックにはアドレスBのデータがステートMで格納されている

CPU#1: アドレスBのデータをステートMで持っている

ブリッジキャッシュ#1: アドレスAのデータをステートMで持っている

CPU#3: アドレスAのデータをステートMで持っている

以下、このような初期状態から、CPU#0がアドレスAをリードするリクエストを発した場合のシステムの動作について説明する。

【0244】(1) CPU#0は、自らのキャッシュを検索するもののアドレスAのデータが存在しないので、上位バス#0にReadトランザクション（アドレスA）を発行する。

【0245】(2) バスブリッジ#0は、上位バス#0から信号線504を経由してトランザクションの種類（Read）及びアドレス（A）を得る。これらの情報は、上位トランザクションバッファ312、タグリード回路318、下位バス出力判定回路314に入力される。すなわち、(2.1) 上位トランザクションバッファ312は、入力された上位バス・トランザクションの種類、アドレス及びスプリットIDなど上位バストランザクションに関する諸情報を格納する。なお、スプリットIDとは、バス上のトランザクションを識別するためのIDであり、トランザクションを発したエージェン

(CPU、バスブリッジ等)のIDと当該エージェント内でのトランザクションのIDを含んでいる。

【0246】すなわち、この動作例では、トランザクション種類“Read”、アドレス“A”及びReadに割り当てられたスプリットIDなどの情報が上位トランザクションバッファ312に格納される。

【0247】(2.2) タグリード回路318は、入力されたアドレスによって指定されるデータがブリッジキャッシュ内に存在するか否かを調べる。このとき、まず、タグリード回路318は、入力されたアドレスからインデックスを抽出し、セクタ350を介してタグRAM402に入力する。タグRAM402は、入力されたインデックスに対応するブロックのタグ情報及びステートをタグリード回路318に返す。

【0248】タグリード回路318は、タグRAM402から返されてきたタグとトランザクション・アドレスのタグ部とを比較し、両者の一致・不一致を判定する。そして、両者が一致する場合には該当ブロックのステートを下位バス出力判定回路314に出力する。この場合、当該ブロックのステートがM又はE又はSであれば、キャッシュRAM404内にReadの要求するデータがあるということを意味する。

【0249】一方、両者が不一致の場合は、キャッシュRAM404内にはReadが要求するデータはないということなので、タグリード回路318はReadの要求するアドレスAに対するブリッジキャッシュのステートとして、ステートIを下位バス出力判定回路314に出力する。ただし、タグが不一致の場合には、該当ブロックに格納されている別アドレスのデータがどのステートであるかによってエビクションが必要となる場合があるので、タグリード回路318は、タグRAM402から受け取った該当ブロック自体のステートも下位バス出力判定回路314に出力する。

【0250】この動作例では、ブリッジキャッシュ#0にはアドレスAに対応するブロックにアドレスBのデータがステートMで格納されているので、タグリード回路318から下位バス出力判定回路314へは、Readの要求データ(アドレスA)が存在しないことを示すステートIの信号と、該当ブロック自体のステートMの信号が出力される。

【0251】(2.3) 下位バス出力判定回路314は、信号線504から入力されたトランザクションの種類と、タグリード回路318から入力されたトランザクション・アドレスのステートとに基づき、下位バスに出力すべきトランザクションを決定する。このとき、下位バス出力判定回路314は、制約1を採用する場合には表3又は4の規則に従い、制約2を採用する場合は表5の規則に従って、下位バスに出力するトランザクションを決定する。

【0252】この動作例では、上位バス・トランザクシ

ョンはReadであり、Readのアドレスに対するブリッジキャッシュ#0のステートはIなので、制約1及び2のいずれを採用した場合でも、下位バス出力判定回路314は上位バスのトランザクションをスプリットし、下位バスに対してアドレスAのデータを要求するReadを発行すると決定する。

【0253】下位バス出力判定回路314の判定結果は、下位バストランザクション生成・保持回路330に入力される。なお、この場合は、上位バスのReadはスプリットされるので、下位バス出力判定回路314は、スプリットを行ったことを示す情報を出力する。この情報は上位バストランザクションバッファ312に前記トランザクションの種類及びアドレス等と対応づけて登録される。また、この情報は、レスポンス生成回路310にも出力される。

【0254】(2.4) 下位バストランザクション生成・保持回路330は、与えられた判定結果に従って、アドレスAのデータを要求するReadトランザクションを生成する。同時に、下位バストランザクション生成・保持回路330は、下位バスアービトレーション回路328に所定の信号を発し、これを受けた下位バスアービトレーション回路328は、下位バスの使用权を得るために信号線514から下位バスに対してアービトレーション信号を発する。この例では各バスは分散アービトレーション方式を採用しており、下位バスアービトレーション回路328は、下位バスアービトレーション回路328が発したアービトレーション信号に対する他のバスブリッジ#1のバス要求状態を信号線512から得、自バスブリッジ#0が下位バスに対してトランザクションを発行できるか否かを判定する。そして、下位バスアービトレーション回路328は、下位バスに対してトランザクションが発行可能となったときに出力バッファ362に対して所定の信号を発し、この信号により、下位バストランザクション生成・保持回路330で生成されたReadが出力バッファ362から信号線516を経由して下位バスに出力される。

【0255】(2.5) このようにしてバスブリッジ#0が下位バスに対して発行したReadは、下位バスから信号線516を経由してバスブリッジ#0自身に取り込まれ、そのトランザクション種類(Read)及びアドレス(A)が下位バストランザクションバッファ326に格納される。

【0256】(2.6) レスポンス生成回路310は、下位バス出力判定回路314からスプリットを行ったことを示す信号を受取り、これに基づきスプリット終了信号を生成し、上位バス#0に出力する。なお、CPU#0は、このスプリット終了信号を受取ると、いったん上位バス#0を解放し、下位バスに発行したReadの結果がバスブリッジ#0から返ってくるのを待つ。

【0257】(3) バスブリッジ#0は、前記(2)の

処理と同時に、ブリッジキャッシュ#0のエビクションが必要か否かを判定する。エビクションが必要と判定された場合、ブリッジキャッシュ#0の該当ブロックのデータをキャッシュRAM404からエビクションバッファ336に退避させる。また、バスブリッジ#0は、エビクションが必要な場合は、上位バス#0へのInvalidateトランザクションを生成し、さらに下位バスへのWriteBackトランザクションの発行の要否を判定する。すなわち、(3.1) 下位バス出力判定回路314は、前記(2.3)におけるタグリード回路318からの出力に基づきエビクションの要否を判定する。エビクションが必要となるのは、トランザクションのアドレスに対応するキャッシュブロックが、他のアドレスによって使用されている（当該他のアドレスについてのステートがM又はE又はSである）場合である。したがって、下位バス出力判定回路314は、トランザクションに対するステートがIであり、該当ブロック自体のステートがM又はE又はSであった場合には、ブリッジキャッシュの該当ブロックのエビクションが必要と判定する。この判定結果は、キャッシュ制御回路322に入力される。なお、下位バス出力判定回路314からキャッシュ制御回路322に与えられる出力には、エビクションすべきブロックのアドレス情報が含まれる。

【0258】この動作例では、初期状態として、ブリッジキャッシュ#0において、アドレスAに対応するブロックにはアドレスBのデータがステートMで格納されている。したがって、下位バス出力判定回路314は、ブリッジキャッシュ#0のエビクションが必要と判定し、アドレスBをキャッシュ制御回路322に出力する。

【0259】(3.2) 下位バス出力判定回路314の出力を受け取ったキャッシュ制御回路322は、その出力に含まれるアドレス情報をキャッシュRAM404に入力する。キャッシュRAM404は、そのアドレス情報に基づき、エビクション対象ブロックのデータをエビクションバッファ336に出力し、当該ブロックを空き状態にする。エビクションバッファ336は、キャッシュRAM404から受け取ったデータを格納する。

【0260】また、キャッシュ制御回路322は、前記アドレス情報をエビクションバッファ336に入力する。エビクションバッファ336は、このアドレス情報を、前記キャッシュRAM404から受け取ったデータに対応付けて記憶する。

【0261】この動作例では、アドレスBに対応するブロックのデータがキャッシュRAM404からエビクションバッファ336に掃き出され、キャッシュRAM404はそのデータをアドレスBと対応付けて格納する。

【0262】この結果、キャッシュRAM404のアドレスBに対応するブロックは解放され、Readトランザクションが使用できる状態となる。

【0263】このように、図6の構成では、エビクシ

ョン対象のブロックをエビクションバッファ336に退避することにより、上位バスから受け取ったトランザクションのために当該ブロックを使用することができる。したがって、この構成では、エビクションに伴うInvalidate及びWriteBackの処理の完了を待たずに、下位バスへのトランザクションの発行等のトランザクションに対する通常処理を進行することが可能となる。

【0264】(3.3) 下位バス出力判定回路314は、エビクションが必要と判定した場合には、タグリード回路318から与えられるエビクション対象ブロックのステート情報に基づき、下位バスにWriteBackを発行するか否かを判定する。この判定は、制約1を採用した場合には表16の規則に従い、制約2を採用した場合には表17の規則に従って行われる。

【0265】この動作例では、ReadのアドレスAに対応するブロックにはアドレスBのデータがステートMで格納されているので、いずれの制約条件を採用する場合でも、下位バスにWriteBackを行うことが決定される。

【0266】この判定結果は、上位バストランザクションバッファ312に入力され、対応トランザクションの情報に対応づけて登録される。

【0267】なお、この判定に基づくWriteBackの発行は、次に説明する上位バス#0へのInvalidateが完了した後に行われる。

【0268】(3.4) 上位バス出力判定回路316は、前記(2.3)におけるタグリード回路318からの出力に基づき、前記下位バス出力判定回路314と同様にして、エビクションの要否を判定する。ここでエビクションが必要と判定された場合は、上位バス出力判定回路316は、エビクション対象のブロックのステート（すなわち、該当ブロックに格納されているデータのステート）に基づき、前記表15の規則に従って上位バスへのInvalidateの発行を決定する。そして、上位バス出力判定回路316は、この決定に応じて、Invalidateの発行を指示する信号を上位バストランザクション生成・保持回路304に出力する。

【0269】この信号を受け取った上位バストランザクション生成・保持回路304は、Invalidateトランザクションを生成すると共に、上位バスアービトレーション回路302に対してアービトレーションを指示する。そして、上位バス#0の使用権が得られ次第、上位バストランザクション生成・保持回路304から出力バッファ360を介して、上位バス#0にInvalidateが発行される。

【0270】この動作例では、上位バス出力判定回路316は、エビクションが必要と判定する。この結果、上位バストランザクション生成保持回路304にてInvalidateが生成され、上位バス#0のエビクシ

ンが完了し次第発行される。したがって、例えばパイプライン処理が可能なシステムとすれば、CPU#0からのReadの発行のサイクルが終わり次第、バスブリッジ#0から上位バス#0に対してInvalidateの発行が可能となる。

【0271】以上に説明した(3)の処理は、前記(2)の処理と同時並行して行われる。そして、これ以降、本システムでは、(2)に続く通常のトランザクション処理(4)～(8)と、(3)に続くエビクション処理(9)～(11)とを行う。本システムにおいてパイプライン方式のアーキテクチャを採用すれば、これら通常処理とエビクション処理とは同時並行的に行うことも可能である。

【0272】(4)バスブリッジ#1は、前記(2)で下位バスに発行されたトランザクションをモニタする。すなわち、バスブリッジ#1は、信号線516から下位バス・トランザクションの種類(Read)及びアドレス(A)を得て、これらの情報を下位トランザクションバッファ326、タグリード回路318、上位バス出力判定回路316に入力する。すなわち、(4.1)下位トランザクションバッファ326は、入力された下位バス・トランザクションの種類、アドレス及び当該トランザクションを発行したバスブリッジのIDなど下位バストランザクションに関する諸情報を格納する。

【0273】この動作例では、トランザクション種類“Read”、アドレス“A”等の情報が下位バストランザクションバッファ326に格納される。

【0274】(4.2)タグリード回路318は、前述のステップ(2.1)と同様にしてタグRAM402を検索し、下位バス・トランザクションの要求データに対するブリッジキャッシュ#1のステートを調べ、そのステートを上位バス出力判定回路316に出力する。

【0275】この例では、ブリッジキャッシュ#1は、Readの要求データをステートMで所有しているので、タグリード回路318からは、ステートMを示す信号が上位バス出力判定回路316に与えられる。

【0276】(4.3)上位バス出力判定回路316は、信号線516から入力されたトランザクション種類とタグリード回路318から入力されたトランザクションに対するステートの情報とから、上位バス#1に発行するトランザクションを決定する。ここで、上位バス#1に発行するトランザクションの決定においては、制約1を採用する場合には表9の規則に従い、制約2を採用する場合には表10の規則に従う。この動作例では、いずれの制約条件を採用した場合でも、Readの発行が決定される。この上位バス出力判定回路316の判定結果は、上位バストランザクション生成・保持回路304に入力される。

【0277】(4.4)上位バストランザクション生成・保持回路304は、与えられた判定結果に従って、アドレ

スAのデータを要求するReadトランザクションを生成する。同時に、上位バストランザクション生成・保持回路304は、上位バスアービトレーション回路302に所定の信号を発し、これを受けた上位バスアービトレーション回路302は、上位バス#1の使用権を得るために信号線500から上位バス#1に対してアービトレーション信号を発する。そして、上位バスアービトレーション回路328は、このアービトレーション信号に対する他のバスブリッジのバス要求状態を信号線502から得、バスブリッジ#1が上位バス#1に対してトランザクションを発行できるか否かを判定する。そして、上位バス#1に対してトランザクションが発行可能となったときに、上位バスアービトレーション回路302は出力バッファ360に対して所定の信号を発し、この結果上位バストランザクション生成・保持回路304で生成されたReadが出力バッファ360から信号線504を経由して上位バス#1に発行される。

【0278】(4.5)このようにしてバスブリッジ#1が上位バスに対して発行したReadは、上位バスから信号線504を経由してバスブリッジ#1自身に取り込まれ、そのトランザクション種類(Read)及びアドレス(A)が上位バストランザクションバッファ312に格納される。

【0279】(5)上位バス#1に接続されたCPU#2及び#3は、バスブリッジ#1から上位バス#1に発行されたReadをモニタする。ここで、CPU#3はキャッシュ内にアドレスAのデータをステートMで持っているので、CPU#3は当該Readに対してHITMをスヌープ出力するとともに、アドレスAに対応するキャッシュブロックのデータを上位バス#1に出力する。

【0280】(6)バスブリッジ#1は、CPU#3から出力されたステートMのデータによって自身のブリッジキャッシュの該当ブロックを更新すると共に下位バスにそのデータを出力し、さらに下位バスに対してHITMをスヌープ出力する。そして、バスブリッジ#1は、自身のブリッジキャッシュの該当ブロックのステートを変更する。すなわち、(6.1)CPU#3から出力されたステートMのデータは、信号線510を経由してバスブリッジ#1に入力される。このデータは、セクタ352を介してキャッシュRAM404に入力されると共に、信号線522を経由して下位バスに出力される。ここで、キャッシュRAM404の該当ブロックのデータ内容は前記CPU#3からのデータによって更新される。このときのキャッシュRAM404内における更新ブロックの特定は次のようにして行われる。すなわち、ステップ(4.5)に示したように、バスブリッジ#1は、自らが上位バスに対して発行したReadを信号線504を経由して取り込むが、この上位バスReadトランザクションのアドレスが下位バス出力判定回路314を

介してキャッシュ制御回路322に入力される。キャッシュ制御回路322は、このアドレス情報からインデックスを抽出し、このインデックス情報をキャッシュRAM404に与えることにより更新すべきブロックを特定する。このようにして特定されたブロックに対して信号線510から入力されたデータが上書きされる。なお、信号線522から下位バスに出力されたデータは、バスブリッジ#0によって取り込まれると共に、メインメモリにも入力され、データ内容の更新が行われる。

【0281】(6.2) バスブリッジ#1から下位バスに対するスヌープ出力は、下位バススヌープ出力生成回路332にて生成される。下位バススヌープ出力生成回路332には、信号線506からの上位バスのスヌープ結果(上位の各CPUのスヌープ出力のワイヤードOR)と、ステップ(4.1)で下位バストランザクションバッファ330に格納された下位バス・トランザクションの種類と、上位バス出力判定回路316がステップ(4.2)で検出したブリッジキャッシュのステートと、が入力される。ここで、制約1を採用した場合、下位バススヌープ出力生成回路332は、下位バストランザクションの種類とブリッジキャッシュのステートとの組み合わせから表11の規則に従ってスヌープ出力を生成する。また、制約2を採用した場合は、下位バススヌープ出力生成回路332は、下位バストランザクションの種類、ブリッジキャッシュのステート及び上位バススヌープ結果の組み合わせから表12の規則に従ってスヌープ出力を生成する。このようにして生成されたスヌープ出力は、信号線518を経由して下位バスに出力される。

【0282】この動作例では、下位バストランザクションの種類がReadで、ブリッジキャッシュのステートがMなので、いずれの制約条件を採用する場合でも、下位バススヌープ出力生成回路332はHITMを出力する。

【0283】(6.3) ブリッジキャッシュの次ステートはタグ更新回路324で決定される。タグ更新回路324には、信号線506からの上位バススヌープ結果と、ステップ(4.5)において上位バストランザクションバッファ312に格納された上位バス・トランザクションの種類が与えられる。タグ更新回路324は、これらの情報に基づき表13又は表14の規則に基づいて、ブリッジキャッシュの次ステートを決定する。この例では、上位バススヌープ結果がHITMで上位バストランザクションがReadなので、ブリッジキャッシュの次ステートはSと決定される。

【0284】また、タグ更新回路324は、上位バストランザクションバッファ312からトランザクションのアドレスAを得、このアドレスAと前記次ステートSとをタグライト回路320に入力する。タグライト回路320は、このアドレスAのインデックス部をタグRAM404に入力してタグ情報の書き換えを行うべきブロッ

クを指定し、さらに前記次ステートSをタグRAM402に入力して当該ブロックのステートを変更する。

【0285】(7) バスブリッジ#0は、下位バスに発行したトランザクションに対する他のバスブリッジからの応答を検出し、上位バス#0に対する応答の生成を行う。この例では、下位バス上にはバスブリッジ#0及びバスブリッジ#1が接続されているだけなので、バスブリッジ#0は、前記(6)でバスブリッジ#1から下位バスに出力されたデータやスヌープ出力などを検出すると、それらに基づき次のような処理を行う。

【0286】まず、バスブリッジ#0は、CPU#0に応答を行うためにスプリット応答トランザクションを生成し、上位バス#0上に発行する。また、バスブリッジ#0は、CPU#0の次ステートを制御するために、スヌープ出力を生成して上位バス#0に出力する。また、バスブリッジ#0は、下位バスから得られたデータをブリッジキャッシュ#0に書き込む。さらに、バスブリッジ#0は、自らのブリッジキャッシュのステートを変更する。すなわち、(7.1) スプリット応答生成回路306は、信号線518から下位バススヌープ結果が入力されると、上位バストランザクション生成・保持回路304に対してスプリット応答トランザクション生成指令を発する。すなわち、スプリット応答生成回路306は、下位バススヌープ結果を、スプリット応答トランザクションを生成するためのタイミング信号として用いている。スプリット応答生成回路306は、下位バススヌープ結果を得たときに、上位トランザクションバッファ312に格納されているトランザクション情報に基づき、トランザクション発行元のCPUに対してスプリット応答トランザクション生成指令を生成する。上位トランザクションバッファ312には、上位バスから受け取ったトランザクションの情報(種類、アドレス、スプリットID等)がスプリットを行ったか否かの情報と共に格納されている。スプリット応答生成回路306は、この上位トランザクションバッファ312からスプリットを行ったトランザクションを検索し、そのトランザクションのスプリットIDからそのトランザクションを発行したCPUのIDを検出して、当該CPU_IDに対するスプリット応答トランザクションを生成する。

【0287】この動作例では、バスブリッジ#1から出力されたHITMの信号が信号線518から入力される。このHITMの信号を受けると、スプリット応答生成回路306は、上位トランザクションバッファ312からスプリットを行ったトランザクションを検出し、そのトランザクションの発行元CPUのID(スプリットIDに含まれている。ここでは#0)を検出し、このCPU#0に対するスプリット応答トランザクションの生成指令を上位バストランザクション生成・保持回路304に出力する。

【0288】上位バストランザクション生成・保持回路

304は、この指令に基づいてスプリット応答トランザクションを生成する。そして、上位バスアービトレーション回路302にて上位バスの使用権を獲得したところで、信号線504から上位バス#0にそのスプリット応答トランザクションが発行される。

【0289】(7.2) バスブリッジ#0は、信号線518から得た下位バススヌープ結果と、前記ステップ(2.5)で下位バストランザクションバッファ326に格納された下位バス・トランザクション種類とに基づき、上位バスに対するスヌープ出力を生成する。このスヌープ出力の生成は、制約1を採用する場合には前記表7の規則に従って行い、制約2を採用する場合には前記表8の規則に従って行う。この処理は上位バススヌープ出力生成回路308で行われ、生成されたスヌープ出力は信号線506を介して上位バス#0に出力される。この例では、スヌープ出力としてHITが上位バス#0に出力される。

【0290】(7.3) バスブリッジ#0は、バスブリッジ#1から下位バスに出力された最新のキャッシュデータを信号線522を経由して受信し、これをキャッシュRAM404に格納する。すなわち、ステップ(7.1)にて上位バス上に発行されたスプリット応答トランザクションはバスブリッジ#0自身にて受信され、このとき下位バス出力判定回路314が上位トランザクションバッファ312に格納されているアドレスを讀出してキャッシュ制御回路322に与える。そして、キャッシュ制御回路322は、このアドレスからインデックスを抽出し、このインデックス情報をキャッシュRAM404に与えることにより書き込み対象のブロックを特定する。そして、キャッシュRAM404では、キャッシュ制御回路322によって指定されたブロックに対して、信号線522から入力されたデータを書き込む。なお、このようにしてキャッシュRAM404に格納された最新データは、続いてキャッシュRAM404から信号線510を介して上位バス#0に出力される。

【0291】この例では、バスブリッジ#0は、下位バスからの応答によって得たアドレスAの最新データをブリッジキャッシュ#0に書き込むと共に、上位バス#1に出力する。

【0292】(7.4) タグ更新回路324は、信号線518から得られる下位バススヌープ結果と、ステップ(2.5)で下位バストランザクションバッファ326に格納された下位バストランザクションの種類とに基づき、ブリッジキャッシュの次ステートを決定する。この次ステートの決定は、制約1及び制約2のいずれを採用する場合でも、前記表6の規則に従って行う。この例では、下位バススヌープ結果がHITMで下位バストランザクションがReadなので、次ステートはSと決定される。

【0293】また、タグ更新回路324は、下位バストランザクションバッファ312から下位バスに発行した

Readトランザクションのアドレスを得、このアドレスと前記次ステートとをタグライト回路320に入力する。タグライト回路320は、このアドレスのインデックス部をタグRAM402に入力して書き換えを行うべきブロックを指定し、さらにアドレスのタグ部及び前記次ステートをタグRAM402に入力して当該ブロックのタグ及びステートを更新する。

【0294】(8) 前記(2)におけるスプリットの結果、いったん上位バス#0を放棄したCPU#0は、バスブリッジ#0から上位バス#0に発行されたスプリット応答トランザクションを受取り、そのスプリット応答トランザクションに付随するデータを、CPUキャッシュの該当ブロックに書き込む。また、このとき、バスブリッジ#0からのスヌープ出力HITによりCPU#0の受け取るスヌープ結果はHITとなり、この結果、CPU#0は当該ブロックのステートをMESIプロトコルにしたがってSに変更する。

【0295】(9) ここからは、前記(3)の処理に続くエビクションの処理について説明する。この(9)～(11)の処理は、必ずしも前記(4)～(8)の処理の後に来るものではなく、両者の処理は並行的に行うこともできる。

【0296】前記ステップ(3.4)にて発行されたInvalidateを受け取った上位バス#0上の各CPUは、当該Invalidateの対象ブロックを無効化する。この例では、InvalidateはアドレスBに対するものであり、CPU#1はアドレスBのデータをステートMで持っている。したがって、CPU#1は、Invalidateに対する応答として、上位バス#0に対してHITMを出力し、アドレスBのブロックのステートMのデータを上位バス#0に出力する。そして、CPU#0は、キャッシュの当該ブロックのステートをMからIに変更する。

【0297】(10) バスブリッジ#0は、Invalidateに対応してCPU#1から書き戻されたデータを信号線510を経由して受取り、このデータによりエビクションバッファ336の内容を更新する。すなわち、エビクションバッファ336には、すでにステップ(3.2)でキャッシュRAM404に格納されていたアドレスBのデータが退避しているが、このデータに対して、CPU#1から書き戻されてきたアドレスBの最新データが上書きされる。

【0298】(11) (11.1) バスブリッジ#0は、前記ステップ(3.3)にて自らが上位バス#0に発行したInvalidateを信号線504を介して取り込む。このInvalidateを受け取った下位バス判定回路314は、上位トランザクションバッファ312に格納されているWriteBack要否の判定結果(前記ステップ(3.3)参照)を讀み出し、これに従ってWriteBack指令信号を生成する。

【0299】この動作例では、ステップ(3.3)にてWriteBackが必要と判定されているので、下位バス出力判定回路314は、アドレスBのデータを書き戻すためのWriteBackの発行を指示する指令信号を生成する。この指令信号は、下位バストランザクション生成・保持回路330に出力する。

【0300】(11.2)これを受け取った下位バストランザクション生成・保持回路330は、アドレスBについてのWriteBackトランザクションを生成する。そして、このWriteBackは、下位バスアービトレーション回路328により下位バスの使用権が取得でき次第、信号線516を経由して下位バスに発行される。

【0301】(11.3)また、下位バス出力判定回路314からのWriteBack指示は、キャッシュ制御回路322にも入力される。この指示を受けたキャッシュ制御回路322は、エビクションバッファ336からアドレスBのデータを出力させる。このデータは、信号線522を経由して下位バスに出力される。

【0302】このようにして、バスブリッジ#0からのWriteBackによって、アドレスBのデータがメインメモリに書き戻される。

【0303】実施形態2. 以上説明した実施形態1は、階層バスシステムにおいて、バスの電氣的な制約を克服して一つのシステム上に接続できるCPUの数を増やすものであった。

【0304】ところで、一つの階層バスシステム上に接続可能なCPUの数は、そのような物理的な制約のみならず論理的な制約を受けることもある。

【0305】すなわち、同時に複数のバストランザクションのスプリットを許容する場合、各トランザクションに対するスプリット応答を正確に発行元のCPUに返すためにトランザクションにIDを持たせる。このIDをスプリットIDと呼ぶ。従来の階層バスシステムでは、このスプリットIDは、図8に示すように次のように定義されていた。

【0306】

【数1】

スプリットID=CPU_ID+トランザクションID
ここで、CPU_IDは、階層バスシステム内の各CPUごとに一意的に定められる。また、トランザクションIDは、各CPU内においてトランザクションごとに一意的に定められる。このようなスプリットIDをみれば、バス上のトランザクションが一意的に特定され、またスプリット応答の宛先が特定される。

【0307】ところが、このようなスプリットIDでは、システム上に接続可能なCPUの個数がCPU_IDのビット数によって制限されてしまう。例えば、CPU_IDに2ビットが割り当てられている場合は、システムに接続可能なCPUの個数は4個に限定される。これがCPU接続個数に対する論理的な制約である。

【0308】従って、実施形態1のごとくシステムの階層化を行い、物理的な制約を回避したとしても、従来のようなスプリットIDの割り当て方では、上記論理的な制約によりシステムに接続可能なCPUの個数が限られてしまう。

【0309】本実施形態2は、このような論理的な制約を回避しようとするものである。この実施形態は、図1に示したバスブリッジを用いた階層型のシステムに適用される。

【0310】本実施形態では、上記論理的な制約を回避するために、各バスブリッジにそれぞれ固有のID(以下、ブリッジIDと呼ぶ)を与え、下位バス上でのトランザクションのスプリットIDを次のように定義する。

【0311】

【数2】下位バススプリットID=ブリッジID+CPU_ID+トランザクションID

したがって、本実施形態における上位バス及び下位バス上のトランザクションのスプリットIDの構成は図9に示すようになる。

【0312】ここで、CPU、バスブリッジ、I/Oなどの各エージェントのIDは、各バスごとに一意的になるように定める。例えば、図1では下位バスに複数のバスブリッジ、メインメモリ、及びI/O装置が接続されているが、それら各エージェントにIDを与える場合には、下位バス内において一意的なIDを与える。また、各上位バスについても各CPU及びバスブリッジに対して上位バス内において一意的なIDを与える。このようなID割り当てを行った場合、バスブリッジに対しては、上位バスについてのIDと下位バスについてのIDの2種類のIDが与えられる。したがって、図9に示す下位バススプリットIDにおけるブリッジIDは、バスブリッジの2つのIDのうち下位バスについてのIDを指す。

【0313】この方式において、上位バスのトランザクションに対するスプリットIDは、その上位バスを発行したCPUのIDと、そのCPUにおける当該トランザクションのIDから定められる。そして、上位バス・トランザクションをスプリットして下位バスにトランザクションを発行した場合、その下位バス・トランザクションには、元の上位バス・トランザクションのCPU_ID及びトランザクションIDに当該スプリットを行ったバスブリッジの下位バスに対するブリッジIDを加えたものを下位バススプリットIDとして与える。このような下位バススプリットIDによれば、各ブリッジから下位バスに発行されたトランザクションを一意的に特定することができる。

【0314】下位バススプリットIDの割り付けはバスブリッジによって行われる。より具体的に、バスブリッジとして図6に示す構成の装置を用いた場合には、下位バススプリットIDの割り付けは、下位バストランザク

ション生成・保持回路330で行われる。

【0315】このようなスプリットID割り付け方式によれば、1つの上位バスに接続可能なCPUの個数は従来同様限定されるものの、システム全体では、上位バスの数を増やすことによってCPUの個数を増やすことが可能となる。

【0316】変形例1. 次に、実施形態2の下位バススプリットID割り当て方式の変形例1について説明する。

【0317】図10は、この変形例1の方式によるスプリットIDの構成を示す説明図である。図に示すように、この変形例では、下位バススプリットIDを割り当てるときに、上位バススプリットIDに単にブリッジIDを付加するのではなく、上位バススプリットIDのビット数を縮退したものを下位バスにおけるトランザクションIDとする。そして、この下位バス・トランザクションIDにブリッジIDを付加することにより、下位バススプリットIDのビット数を上位バススプリットIDのビット数と等しくする。ここで、上位バススプリットIDから下位バスにおけるトランザクションIDを生成する際には、この下位バス・トランザクションIDがスプリットを行ったバスブリッジにおいてトランザクションごとに一意的となるようにする。

【0318】このように下位バススプリットIDを定めることにより、下位バスに対して上位バス用のデバイス（例えばCPU）を接続することが可能となる。

【0319】なお、この構成においては、上位バスのスプリットIDとこれを圧縮したトランザクションIDとの対応関係を管理するID管理手段をバスブリッジに設ける。変形例2. 次に、実施形態2の下位バススプリットID割り当て方式の変形例2について説明する。

【0320】図11は、この変形例2の方式によるスプリットIDの構成を示す説明図である。図に示すように、この変形例では、ブリッジIDをCPU_IDと同一ビット数になるように設定する。そして、下位バススプリットIDを割り当てるときに、上記変形例1と同様、上位バススプリットIDを変形して下位バスにおけるトランザクションIDを生成する。このとき、本変形例では、上位バスでのトランザクションIDのビット数と等しくなるように上位バススプリットIDを縮退して、下位バス・トランザクションIDを生成する。そして、この下位バス・トランザクションIDにブリッジIDを付加することにより、下位バススプリットIDのビット数を上位バススプリットIDのビット数と等しくする。

【0321】この変形例2の方式によれば、上位バススプリットIDと下位バススプリットIDとが同一ビット数となるだけでなく、上位バススプリットIDにおけるCPU_IDと下位バススプリットIDにおけるブリッジIDとが同一ビット数となる。したがって、この方式によれば、バスブリッジとCPUとを論理的に等価に扱うこ

とができ、図1に示した下位バスに対してCPUを接続することが可能となり、システム構成の柔軟性が向上する。

【0322】なお、以上では、バスブリッジから下位バスに発行するトランザクションに対して与えるスプリットIDの生成方式について説明したが、この変形例の方式はバスブリッジから上位バスに発行するトランザクションに与えるスプリットIDにも応用することができる。この場合、バスブリッジは、下位バスから受け取ったトランザクションのスプリットIDを変形してトランザクションIDを生成し、このトランザクションIDに対して当該バスブリッジの上位バスにおけるブリッジIDを付加することにより上位バススプリットIDを生成する。

【0323】この変形例2における具体的なスプリットIDの割り付け方について説明する。まず、上位バスに発行されたトランザクションをバスブリッジがスプリットして下位バスにトランザクションを発行する場合におけるスプリットIDの割り付け方から説明する。

【0324】この例では、バスブリッジ内に図12に示すような管理テーブルを設け、この管理テーブルを用いて下位バススプリットIDを決定する。図12に示す管理テーブルは、トランザクションIDの数の分だけの欄から構成される。例えば、トランザクションIDが8ビットである場合には、管理テーブルは256の欄から構成される。管理テーブルの各欄には、当該トランザクションIDが使用中であるか未使用であるかを示す使用／未使用フラグが設けられる。

【0325】上位バスのトランザクションをスプリットすると決定した場合、バスブリッジは、図12の管理テーブルから未使用の欄を探し出し、この結果見付かった未使用欄のトランザクションIDを、下位バスに対して発行するトランザクションのトランザクションIDとする。次に、このトランザクションIDに当該バスブリッジのブリッジIDを付加して下位バススプリットIDとする。そして、バスブリッジは、このようにして求められた下位バススプリットIDをヘッダ等に付加して、下位バスに対してトランザクションを発行する。また、このとき、管理テーブルの当該トランザクションIDの欄には、スプリットを行った上位バス・トランザクションのスプリットIDが登録され、使用／未使用フラグが「未使用」から「使用中」に変更される。そして、「使用中」となった欄は、スプリットしたトランザクションに対するスプリット応答トランザクションを発行する際に、使用／未使用フラグが「未使用」に変更される。これにより、その欄に対応するトランザクションIDは再び使用可能となる。

【0326】例えば、バスブリッジとして図6に示す構成を用いた場合、この管理テーブルは上位バストランザクションバッファ312に格納される。上位トランザク

ションバッファ312は、信号線504から入力されるスプリットIDと下位バス出力判定回路314からのスプリットについての情報に基づいて前述したような管理テーブルのメンテナンスを行う。すなわち、バスブリッジが上位バスからのトランザクションを受け取ったとき、その上位バストランザクションのスプリットIDが信号線504を経由して上位トランザクションバッファ312に入力される。そして、下位バス出力判定回路314にてトランザクションのスプリットが決定されると、この決定を示す信号が上位トランザクションバッファ312に入力される。すると、上位バストランザクションバッファ312は、管理テーブルの未使用欄を探し、見付かった未使用欄に上位バスのスプリットIDを格納し、その欄のフラグを「使用中」に変更する。同時に、上位バストランザクションバッファ312は、その欄の番号（トランザクションIDに対応）を示す信号を下位バストランザクション生成・保持回路330に与える。下位バストランザクション生成・保持回路330は、受け取った欄番号に対応するトランザクションIDを求め、このトランザクションIDに当該バスブリッジの下位バス用のブリッジIDを付加して下位バススプリットIDを生成する。

【0327】以上説明したように、図12に示すような管理テーブルを用いて各トランザクションIDの使用／未使用を管理し、上位バストランザクションをスプリットする場合には、それらトランザクションIDのうち未使用のものを順に下位バススプリットID中のトランザクションIDとして使用することにより、下位バスに対して規定されているトランザクションIDを無駄なく使用することが可能になる。すなわち、例えば下位バスにおけるトランザクションIDを、上位バススプリットIDから関数などを用いて定める場合、異なった上位バススプリットIDが同じ下位バストランザクションIDにマッピングされる可能性がある。この場合、後から来た上位バストランザクションはスプリットすることができず、後の上位バストランザクションについてはCPUに対してリトライをさせることになる。これに対して本構成のごとく管理テーブルを用いた場合では、異なった上位バストランザクションに対して同一の下位バストランザクションIDが割り当てられることはなく、トランザクションIDを無駄なく使用することができる。また、リトライの発生頻度を減らすことができるので、システム全体のスループット、レイテンシを向上させることができる。

【0328】なお、この方式においては、リトライが生じるのは管理テーブルに未使用の欄がなくなった場合のみである。このような場合において、上位バスからトランザクションが来た場合は、バスブリッジは上位バスにリトライ終了信号を返し、CPUに同一トランザクションの再発行を要求する。

【0329】次に、下位バスのトランザクションを受け取ったバスブリッジが上位バスにトランザクションを発行する場合の、上位バストランザクションに対するスプリットIDの割り付け方について説明する。

【0330】この場合は、バスブリッジ内に図13に示すような管理テーブルを設け、この管理テーブルを用いて上位バススプリットIDを決定する。図13の管理テーブルは、図12に示したものと同様の構成を有している。

【0331】バスブリッジが、下位バスのトランザクションを受けて上位バスに対してトランザクションを発行する場合、前記図12の場合と同様にして、図13の管理テーブルの未使用欄のトランザクションIDを、上位バスに対して発行するトランザクションのトランザクションIDとする。次に、このトランザクションIDに当該バスブリッジの上位バスに対するブリッジIDを付加して上位バススプリットIDとする。このとき、管理テーブルの当該トランザクションIDに対応する欄には、下位バスバス・トランザクションのスプリットIDが登録され、使用／未使用フラグが「未使用」から「使用中」に変更される。そして、「使用中」の欄は、上位バスに発行したトランザクション終了したときに「未使用」に変更される。これにより、その欄に対応するトランザクションIDは再び使用可能となる。

【0332】例えば、バスブリッジとして図6に示す構成を用いた場合、この管理テーブルは下位バストランザクションバッファ326に格納される。下位バストランザクションバッファ326は、信号線516から入力される下位バススプリットIDと、上位バス出力判定回路316から入力される上位バスに対するトランザクション発行の有無についての情報とに基づき、管理テーブルをメンテナンスする。すなわち、上位バス出力判定回路314にて上位バスに対してトランザクションを発行すると決定された場合、下位バストランザクションバッファ326の管理テーブルの未使用欄に元の下位バストランザクションのスプリットIDが登録され、その欄の番号を示す信号が上位バストランザクション生成・保持回路304に入力される。そして、上位バストランザクション生成・保持回路304は、その欄番号に対応するトランザクションIDを求め、このトランザクションIDに当該バスブリッジの上位バス用のブリッジIDを付加して上位バススプリットIDを生成する。

【0333】このように、図13に示す管理テーブルを用いて上位バススプリットIDを管理することにより、異なった下位バススプリットIDに対して同一の上位バススプリットIDが割り付けられることがなくなり、トランザクションIDを無駄なく使用することができる。したがって、上位バススプリットIDに空きがない場合に起こる下位バストランザクションのウェイト（wait：上位バスにトランザクションが発行可能となるまで

下位バスのトランザクションを待たせること。この間下位バスは当該トランザクションによって占有される) 処理の頻度も減らすことができ、システム全体のスループット、レイテンシを向上させることができる。

【0334】なお、この方式において下位バストランザクションのウェイトが生じるのは、管理テーブルに未使用欄がなくなった場合のみである。

【0335】実施形態3. 実施形態1においては、階層バスシステムにおける上位バスの仕様と下位バスの仕様との関係については特に規定していなかった。

【0336】これに対して本実施態は、上位バスの仕様と下位バスの仕様とに一定の関係を規定することにより、階層バスシステムのシステム構成の柔軟性を向上させることを目的とするものである。

【0337】すなわち、本実施形態では、バスの物理的仕様及び論理的仕様の両方について、上位バスと下位バスとで仕様の統一を図る。ここで、バスの物理的仕様には、バス上の信号の電気レベルやバスが動作可能なクロック周波数などがある。また、論理的仕様としては、バス上のトランザクションの種類及びそのビットパターン、トランザクションの要求アドレスの範囲及びその表現形式、バスに接続されるキャッシュが出力するスヌープ出力の表現形式及びそのビットパターン、バス上におけるトランザクションの識別子であるスプリットIDの構成などがある。なお、スプリットIDの構成の仕様の統一には、前記実施形態2の変形例2において説明したスプリットID構成方式、すなわち上位バススプリットIDと下位バススプリットIDとのビット幅を同一にし、上位バススプリットIDにおけるCPU_IDと下位バススプリットIDにおけるブリッジIDとのビット幅を同一にすることが含まれる。

【0338】このように上位バスと下位バスとの仕様を同一にすることにより、以下に示すようにシステム構成のバリエーションを広げることができる。

【0339】まず、図14に示すように、下位バスに対して、上位バス上のCPU#0~#2と同等のCPU#3を接続可能となる。このとき、下位バスに接続されたバスブリッジ及びCPU#3にはそれぞれ下位バス上で一意的なIDを与え、これをスプリットID生成の際に用いる。具体的なスプリットIDの生成方式としては、実施形態2のものを用いることができる。

【0340】また、同一バス上にCPUとバスブリッジが接続可能という点を拡張すれば、バス上のCPUをバスブリッジと置き換えることにより、図15に示すような多階層のシステムの構築が可能となる。この場合、各バスに接続されたエージェント(CPU、バスブリッジ)には、当該バス上において一意的なIDが与えられ、このIDにより当該バス上におけるトランザクションのスプリットIDを生成する。具体的なスプリットIDの生成方式としては、実施形態2のものを用いること

ができる。なお、図15において、第2上位バスのCPUをバスブリッジに置き換えることにより、さらに多階層のシステムを構築することもできる。

【0341】また、本実施形態において、実際のシステム構成の柔軟性を向上させるためには、モジュール化の手法を採用することもできる。

【0342】すなわち、コンピュータ本体には、メインメモリ及びI/O装置等を接続した下位バス構成のみを設けておき、CPU等の上位の構成はボード化して適宜交換可能とする。交換ユニットたるボードとしては、一つのCPUのみを実装したボードや、1つ以上のCPUとバスブリッジ及びブリッジキャッシュとを1本のバスに接続して構成した1つのクラスタを実装したボード等を用いることができる。これら各種のボードは、下位バスに接続可能なインタフェースを有する。

【0343】この方式によれば、例えば図16に示すように、1つCPUのボード100を接続した1CPUのシステムから、クラスタを収容したボード102を接続したマルチCPUのシステムに容易にバージョンアップが可能となる。このように、この方式によれば、ユーザ等の要望に合わせてシステム構成を容易に変更することが可能となる。

【0344】また、図17に示すように、CPU、メインメモリ及びI/O装置を含む基本的なシステムに対して、追加ボードとしてクラスタを収容したボード102を接続するという構成も可能である。

【0345】更には、下位バスに対して複数のボードを並列接続することにより、システムの性能を更に向上させることも可能である。

【0346】実施形態4. 以上説明した各実施形態は、複数のCPUが1つの下位バスを介して1つのメインメモリを共有するシステムについてのものであった。これに対して、本実施形態は、下位バスーメインメモリを複数系統設け、これらメインメモリをそれぞれ上位の複数のCPUで共有する。

【0347】図18は、実施形態4に係る階層バスシステムの一例の構成を示す説明図である。図18において、図1と同一の部分には同一の番号を付してその説明を省略する。

【0348】図18に示すシステムは、2つの下位バス40a、40bを有し、下位バス40aにはメインメモリ50a及びI/O装置52aが、また下位バス40bにはメインメモリ50b及びI/O装置52bが、それぞれ接続されている。そして、このシステムにおいては、各上位バスには、それぞれ2つ下位バスにそれぞれ接続するために、2系統のバスブリッジ及びブリッジキャッシュが設けられている。すなわち、上位バス20には、下位バス40aと接続するためのバスブリッジ30a及びブリッジキャッシュ31aと、下位バス40bと接続するためのバスブリッジ30b及びブリッジキャッ

シュ 3 1 b とが接続されている。また、上位バス 2 2 には、下位バス 4 0 a と接続するためのバスブリッジ 3 2 a 及びブリッジキャッシュ 3 3 a と、下位バス 4 0 b と接続するためのバスブリッジ 3 2 b 及びブリッジキャッシュ 3 3 b とが接続されている。

【0349】この構成において、メインメモリ 5 0 a 及びメインメモリ 5 0 b にはそれぞれ異なったアドレス空間が割り当てられる。したがって、各バスブリッジは、自身が接続されている下位バスのメインメモリのアドレス空間を自らの受け持ち範囲とする。すなわち、各バスブリッジは、上位バスから受け取ったトランザクションの要求アドレスが自らの受け持ち範囲か否かを判定する機構を有し、受け持ち範囲であると判定された場合のみ、その上位バス・トランザクションに対する応答処理を行う。例えば、CPU 1 0 から発せられたトランザクションがメインメモリ 5 0 b のアドレスに対するものであった場合、バスブリッジ 3 0 a、3 0 b は共にそのトランザクションを受け取ってアドレスが各自の受け持ち範囲に含まれるか否かを判定する。この結果、バスブリッジ 3 0 b がそのトランザクションに対する処理を行うことになる。そして、バスブリッジ 3 0 b は、ブリッジキャッシュ 3 1 b のスヌープを行い、この結果に基づいて必要に応じてトランザクションのスプリットなどの処理を行う。

【0350】したがって、この構成では、上位バス 2 0、2 2 に接続された各 CPU 1 0 ~ 1 5 は、下位バス及びメインメモリが 2 系統存在することを意識せずにトランザクションを発行することができる。

【0351】このように、本実施形態では、システムに複数のメインメモリを設け、各メインメモリを別々の下位バスに接続したことにより、各下位バスの負荷を低減し、下位バスのスループットを向上させることができる。

【0352】なお、上記構成において、各バスブリッジの受け持ち範囲を、コンピュータ起動時のセルフテストにおいて判明した各メインメモリの使用可能アドレス範囲に基づき調節することもできる。この場合、各バスブリッジは、セルフテストで判明した使用可能アドレス範囲情報を得て、各自の接続されている下位バスのメインメモリの使用可能アドレス範囲を自らの受け持ちアドレス範囲として設定する。このような方法によれば、各バスブリッジの受け持ち範囲を最適化することができる。

【0353】また、図 1 8 の例では、各下位バス 4 0 a、4 0 b にそれぞれ I/O 装置 5 2 a 又は 5 2 b が接続されていたが、そのような構成では、トランザクションのアドレスによって使用可能な I/O 装置が限定される。すなわち、あるトランザクションが、メインメモリ 5 0 a のアドレスに対するものの場合、そのトランザクションは I/O 装置 5 2 a のみしか使用できない。逆にいえば、すべてのトランザクションが、アドレスにか

わりなく同一の I/O 装置を利用できるようにするためには、複数の下位バスにそれぞれ同一の I/O 装置を接続する必要がある。

【0354】図 1 9 は、このような問題を解決するためのシステム構成を示す説明図である。図 1 9 において、図 1 8 と同一の部分には同一の符号を付してその説明を省略する。

【0355】図 1 9 の構成では、1 つの I/O 装置 5 2 が 2 つの下位バス 4 0 a、4 0 b によって共有される。すなわち、I/O 装置 5 2 は I/O バス 5 6 に接続され、下位バス 4 0 a 及び 4 0 b はそれぞれバスブリッジ 5 4 a 及び 5 4 b を介して I/O バス 5 6 に接続されている。バスブリッジ 5 4 a 及び 5 4 b にはブリッジキャッシュは設けられない。バスブリッジ 5 4 a 及び 5 4 b は、下位バス 4 0 a 又は 4 0 b から受け取ったトランザクションが I/O 装置 5 2 に対するトランザクションであると判定した場合は、そのトランザクションを I/O バス 5 6 に転送する。そして、I/O 装置 5 2 は、I/O バス 5 6 からトランザクションを受け取り、必要な処理を行う。

【0356】図 1 9 の構成によれば、各トランザクションは、アドレスにかかわらず、システムが有するすべての I/O 装置を使用することができる。

【0357】なお、図 1 9 の構成において、バスブリッジ 5 4 a、5 4 b が、下位バス 4 0 a 又は 4 0 b から I/O バス 5 6 に対してトランザクションを転送するときに、スプリットを行うようにしてもよい。この場合、I/O トランザクション処理時における下位バスの負荷を低減することができる。

【0358】また、バスブリッジとブリッジキャッシュとを 1 つのパッケージとして構成し、ブリッジキャッシュを使用するモードと使用しないモードとを切り換え可能とすることにより、一種類のバスブリッジ・パッケージで上位バスー下位バス間用と下位バスー I/O バス間用の両方に用いることができる。

【図面の簡単な説明】

【図 1】 本発明に係る階層バスシステムの全体的な構成の一例を示す概略構成図である。

【図 2】 実施形態 1 における変形例の概略構成を示す説明図である。

【図 3】 エビクションバッファを設けたバスブリッジの概略構成を示す説明図である。

【図 4】 補助キャッシュを設けたバスブリッジの概略構成を示す説明図である。

【図 5】 リトライ判定機能を有するバスブリッジの概略構成を示す説明図である。

【図 6】 実施形態 1 におけるバスブリッジ及びブリッジキャッシュの構成を示すブロック図である。

【図 7】 階層バスシステムの一例を示す図である。

【図 8】 従来のスプリット ID の構成を示す図であ

る。

【図9】 実施形態2における上位バスブリットIDと下位バスブリットIDの構成を示す図である。

【図10】 実施形態2の変形例1における上位バスブリットIDと下位バスブリットIDの構成を示す図である。

【図11】 実施形態2の変形例2における上位バスブリットIDと下位バスブリットIDの構成を示す図である。

【図12】 実施形態2の変形例2における下位バスランザクションIDの管理テーブルを示す図である。

【図13】 実施形態2の変形例2における上位バスランザクションIDの管理テーブルを示す図である。

【図14】 実施形態3における下位バスにCPUを接続した階層バスシステムの構成を示す図である。

【図15】 実施形態3における多階層の階層バスシステムの構成を示す図である。

【図16】 実施形態3におけるボード交換を示す説明図である。

【図17】 実施形態3におけるボード追加を示す説明図である。

【図18】 実施形態4に係る階層バスシステムの一例の概略構成を示す説明図である。

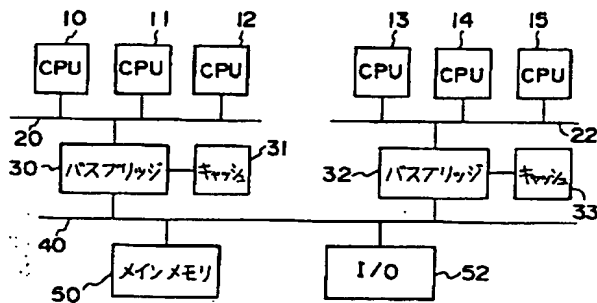
【図19】 実施形態4において複数の下位バスがI/O装置を共有する構成例を示す説明図である。

【図20】 バス共有型の密結合マルチプロセッサシステムを示す概略図である。

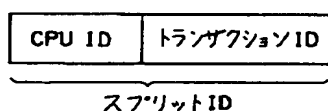
【符号の説明】

10～15 CPU、20、22 上位バス、30、32 バスブリッジ、31、33 ブリッジキャッシュ、40 下位バス、50 メインメモリ、52 I/O装置、60 バスブリッジ制御部、62 エビクションバッファ、64 アドレスレジスタ、66 データバッファ、68 補助キャッシュ、72 スプリット情報レジスタ、80 上位キャッシュ状態フラグ記憶部、300 バスブリッジ、302 上位バスアービトレーション回路、304 上位バスランザクション生成保持回路、306 スプリット応答生成回路、308 上位バススヌープ出力生成回路、310 レスポンス生成回路、312 上位バスランザクションバッファ、314 下位バス出力判定回路、316 上位バス出力判定回路、318 タグリード回路、320 タグライト回路、322 キャッシュ制御回路、324 タグ更新回路、326 下位バスランザクションバッファ、328 下位バスアービトレーション回路、330 下位バスランザクション生成保持回路、332 下位バススヌープ出力生成回路、334 レスポンス受信回路、336 エビクションバッファ、338 リトライ判定回路、402 タグRAM、404 キャッシュRAM。

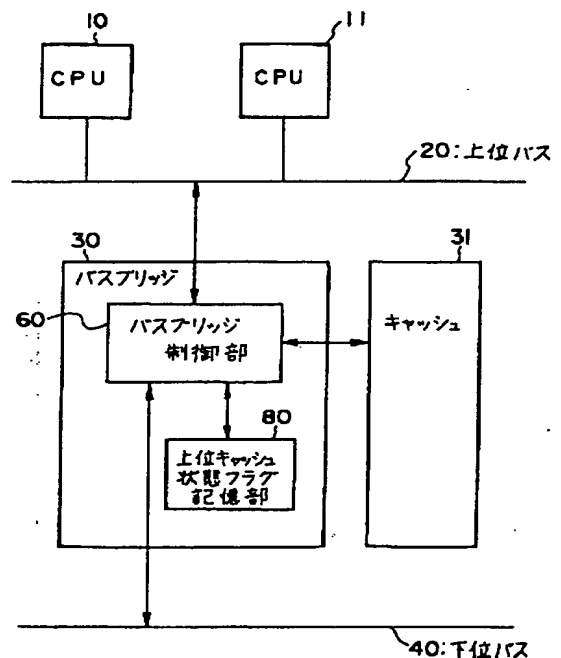
【図1】



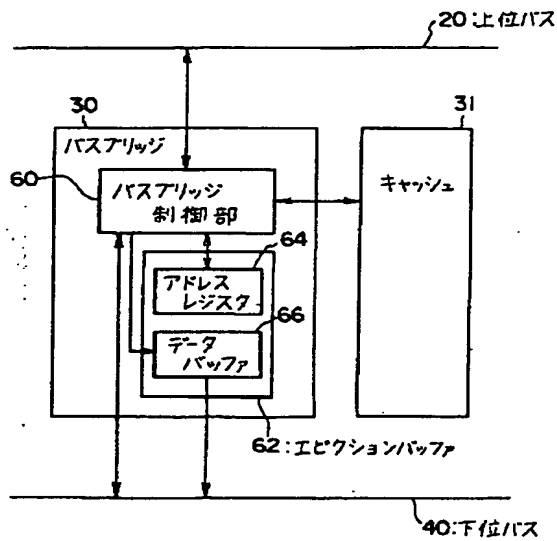
【図8】



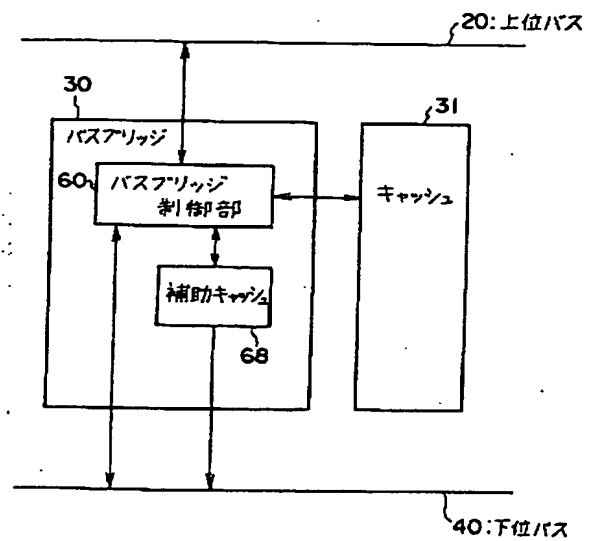
【図2】



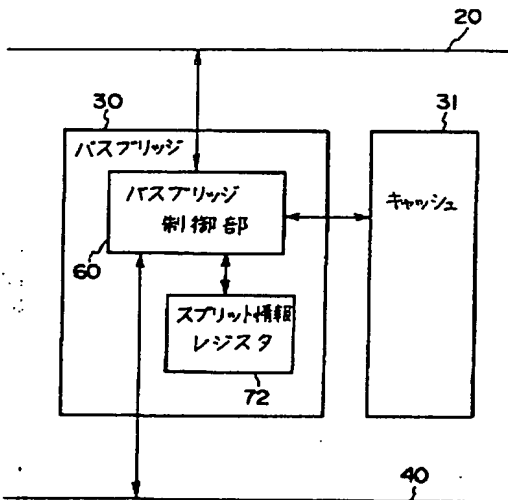
【図 3】



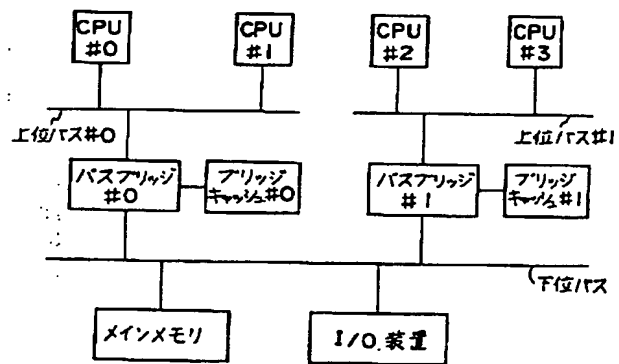
【図 4】



【図 5】

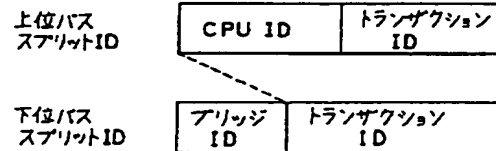
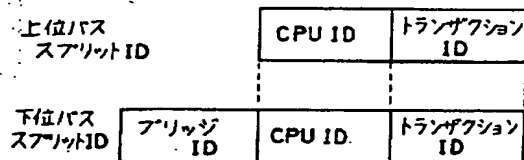


【図 7】

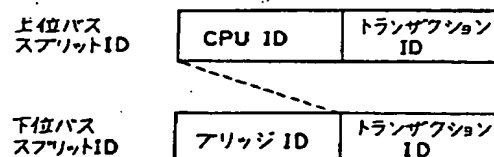


【図 10】

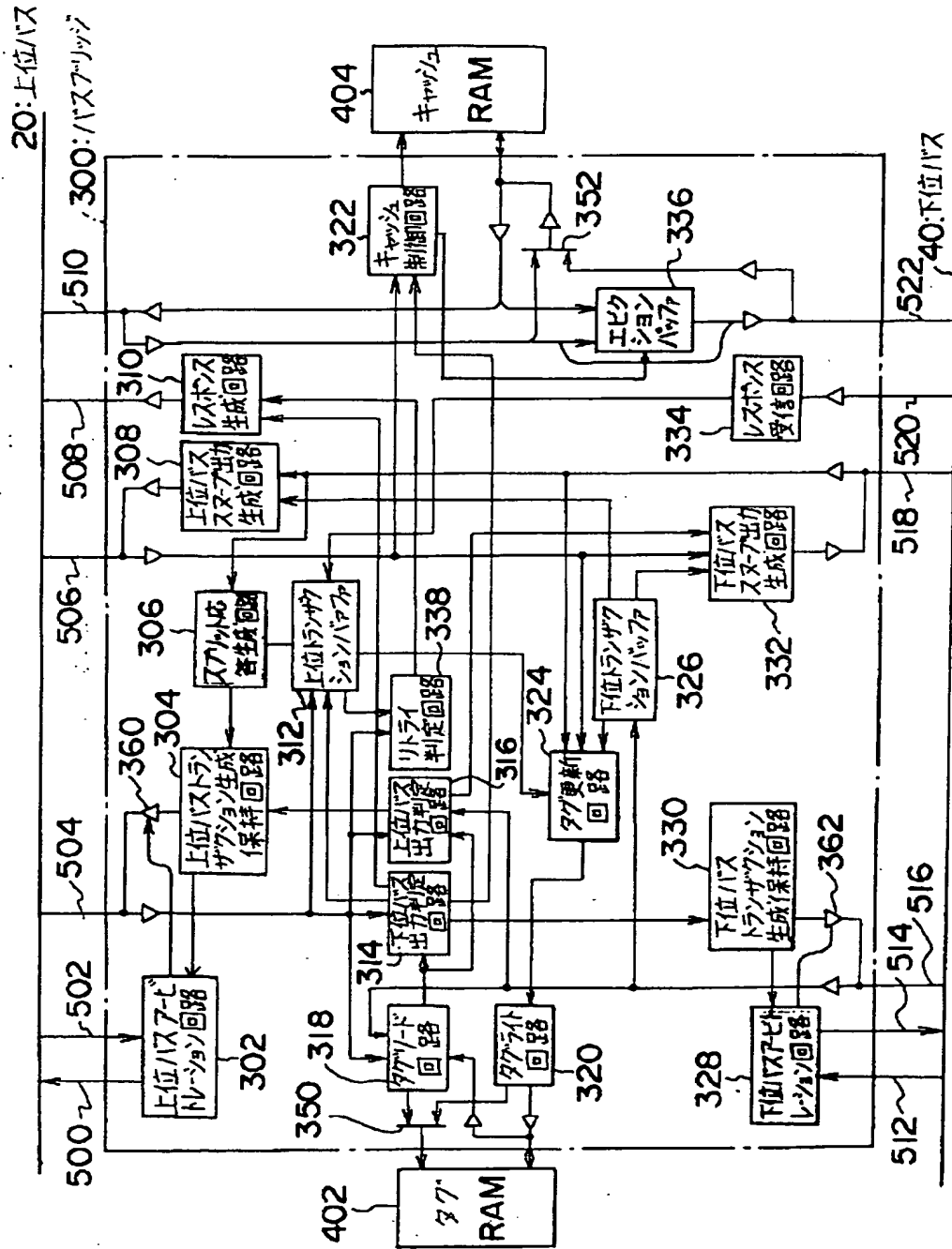
【図 9】



【図 11】



【図6】



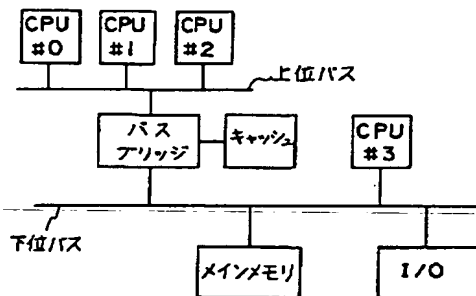
【図12】

下位バス トランザクションID	上位バス スプリットID	使用/未使用フラグ
0	30	使用中
1	01	使用中
2		未使用
3		未使用

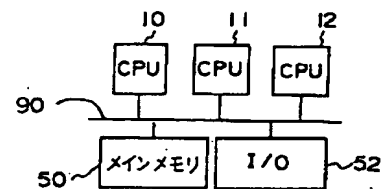
【図13】

上位バス トランザクションID	下位バス スプリットID	使用/未使用フラグ
0	21	使用中
1	02	使用中
2		未使用
3		未使用

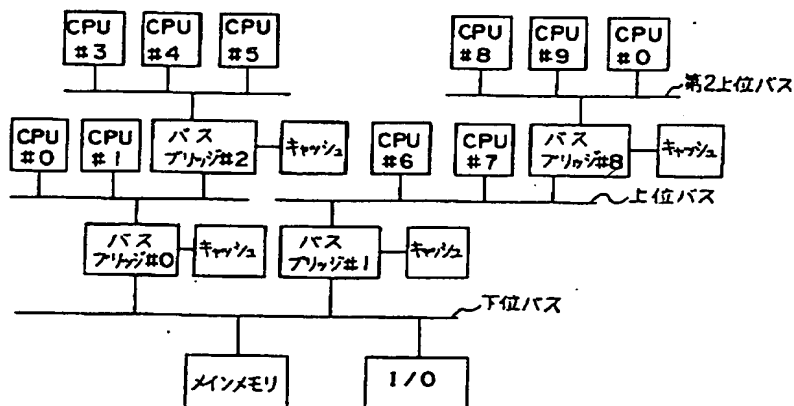
【図14】



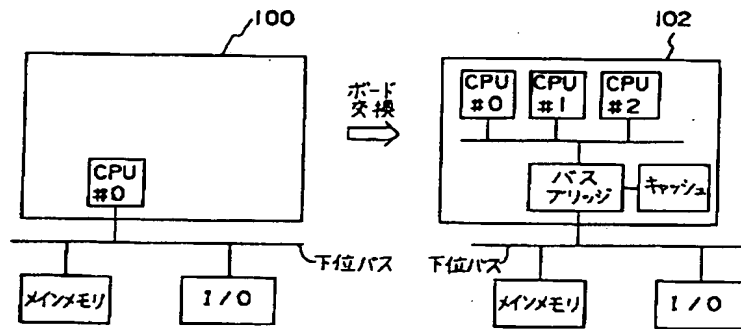
【図20】



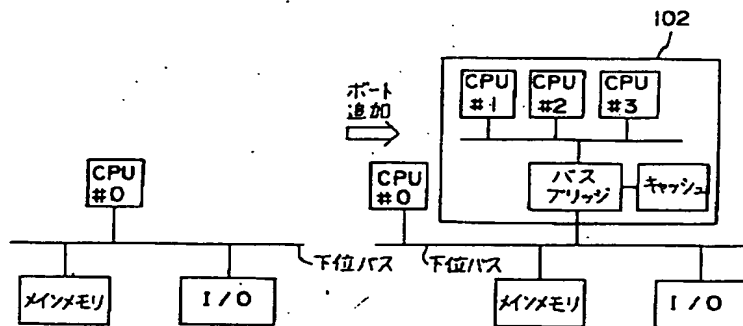
【図15】



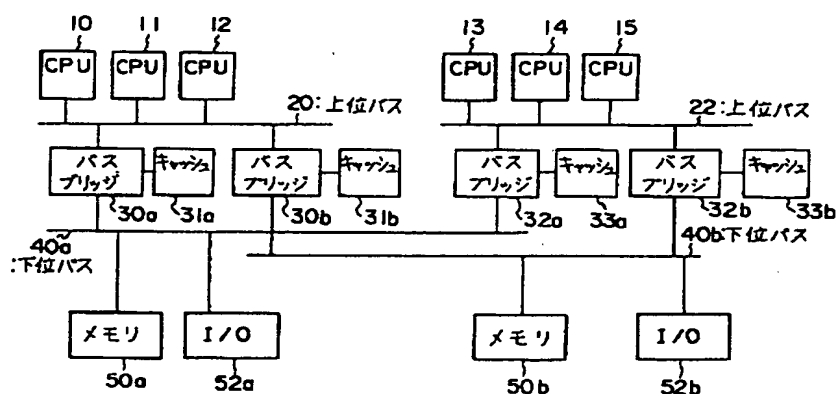
【図 16】



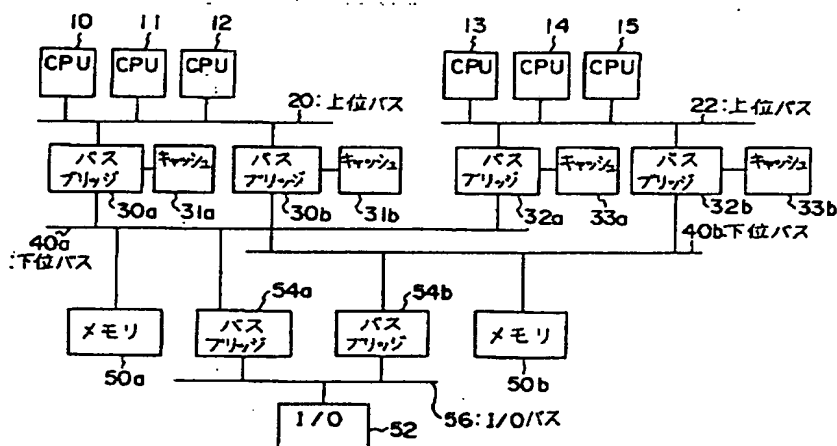
【図 17】



【図 18】



【図 1-9】



【手続補正書】

【提出日】平成 7 年 1 1 月 6 日

【手続補正 1】

【補正対象書類名】明細書

【補正対象項目名】0 3 1 8

【補正方法】変更

【補正内容】

【0 3 1 8】このように下位バススプリット ID を定めることにより、下位バスに対して上位バス用のデバイス（例えば CPU）を接続することが可能となる。なお、この構成においては、上位バスのスプリット ID とこれ

を圧縮したトランザクション ID との対応関係を管理する ID 管理手段をバスブリッジに設ける。

【手続補正 2】

【補正対象書類名】明細書

【補正対象項目名】0 3 1 9

【補正方法】変更

【補正内容】

【0 3 1 9】変形例 2. 次に、実施形態 2 の下位バススプリット ID 割り当て方式の変形例 2 について説明する。

**This Page is Inserted by IFW Indexing and Scanning
Operations and is not part of the Official Record**

BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images include but are not limited to the items checked:

- ☐ BLACK BORDERS
- ☐ IMAGE CUT OFF AT TOP, BOTTOM OR SIDES
- ☐ FADED TEXT OR DRAWING
- ☒ BLURRED OR ILLEGIBLE TEXT OR DRAWING
- ☐ SKEWED/SLANTED IMAGES
- ☐ COLOR OR BLACK AND WHITE PHOTOGRAPHS
- ☐ GRAY SCALE DOCUMENTS
- ☐ LINES OR MARKS ON ORIGINAL DOCUMENT
- ☐ REFERENCE(S) OR EXHIBIT(S) SUBMITTED ARE POOR QUALITY
- ☐ OTHER: _____

IMAGES ARE BEST AVAILABLE COPY.

As rescanning these documents will not correct the image problems checked, please do not report these problems to the IFW Image Problem Mailbox.

THIS PAGE BLANK (USPTO)